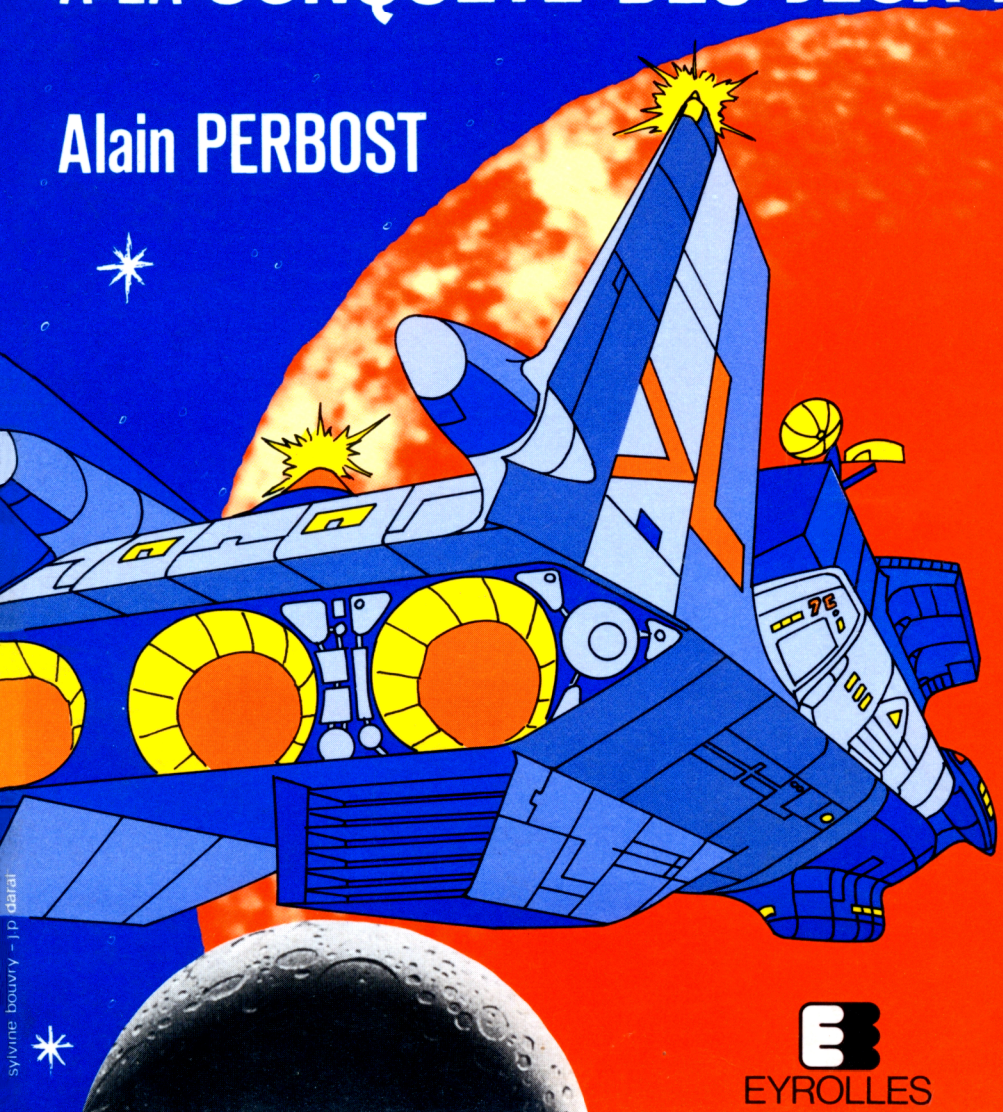
MICRO-ORDINATEURS

ZX SPECTRUM

A LA CONQUÊTE DES JEUX !

Alain PERBOST



EYROLLES

sylvine bouvry - j.p darai

ZX SPECTRUM
A LA CONQUÊTE DES JEUX

Dessins de Eddie PELLEGRIN

« La loi du 11 mars 1957 n'autorisant, aux termes des alinéas 2 et 3 de l'article 41, d'une part, que les «copies ou reproductions strictement réservées à l'usage privé du copiste et non destinées à une utilisation collective» et, d'autre part, que les analyses et les courtes citations dans un but d'exemple et d'illustration, « toute représentation ou reproduction intégrale, ou partielle, faite sans le consentement de l'auteur ou de ses ayants droit ou ayants cause, est illicite » (alinéa 1 de l'article 40) ».

« Cette représentation ou reproduction, par quelque procédé que ce soit, constituerait une contrefaçon sanctionnée par les articles 425 et suivants du Code pénal ».

© Editions EYROLLES, 1984

ZX SPECTRUM
A LA CONQUÊTE DES JEUX

par

Alain PERBOST

Collection animée
par Richard SCHOMBERG

EYROLLES

61, boulevard Saint-Germain — 75005 Paris
1985

DANS LA MÊME COLLECTION

SCHOMBERG	- Le Basic Universel.
SCHOMBERG	- Micro-ordinateurs : Comment ça marche ?
HERNANDEZ	- Pascal par l'exemple.
NOLLET	- La conduite du ZX 81.
PELLIER	- La conduite du TRS 80.
LADEVIE	- Votre gestion avec BASIC sur micro-ordinateur.
QUEINNEC	- Langage d'un autre type : LISP.
PELLIER	- Programmez vos jeux d'action rapide sur TRS 80.
ASTIER	- La conduite de l'APPLE II. Tome 1 : le Basic de l'APPLE II. Tome 2 : le système graphique et l'assembleur de l'APPLE II.
MONTEIL	- L'assembleur facile du 6502 et du 6510.
LEPAPE	- L'assembleur facile du Z 80.
OROS et PERBOST	- ZX 81 à la conquête des jeux.
	- CASSETTE - ZX 81 à la conquête des jeux.
PERBOST	- CASSETTE N° 2 - 13 jeux 1 K.
DAX	- CP/M et sa famille.
NOLLET	- Langage machine, trucs et astuces sur ZX 81.
BICKING	- La conduite du PC 1212 (ou TRS 80 pocket).
TEJA	- Apprenez à parler à votre ordinateur.
MONTEIL	- La conduite du VIC 20.
PLOUIN	- La conduite de l'IBM-PC.
SAGUEZ	- Télécommande avec votre micro-ordinateur.
PELLIER	- Tout sur les disques du TRS 80 modèles I et III.
OROS et PERBOST	- La conduite du FX-702 P.
GROS	- La conduite du PC 1500.
HARWOOD	- Jeux et applications pour ZX SPECTRUM.
HARTNELL	- Le grand livre du ZX SPECTRUM.
HARTNELL et JONES	- La conduite du ZX SPECTRUM.
VULDY	- Graphisme 3 D sur votre micro-ordinateur.
BOUQUEROD	- Des extensions à construire pour votre ZX 81.
PINSON	- Le Basic en gestion sur Apple II.
WILLARD	- La conduite du TI 99.
CEYRAT	- Mon TI 99/4A.
DELANNOY	- Les fichiers en Basic sur micro-ordinateur.
AUBERT	- Pratiquez l'intelligence artificielle.
PERBOST et MASSE	- VIC 20 à la conquête des jeux.
TERRAL	- La conduite du T 07.
ASTIER	- La conduite de l'ORIC-1.
MONTEIL	- Premiers pas en LOGO.
MONTEIL	- La conduite du COMMODORE 64. Tome 1 : Basic, graphisme et son. Tome 2 : Langage-machine entrées/sorties et périphériques.
OROS	- La conduite de l'ATARI 400/800.
WILLARD	- TI 99 à la conquête des jeux.
KRUTCH	- Expériences d'intelligence artificielle en Basic.

ASTIER	- ORIC-1 à la conquête des jeux.
PELLIER	- Langage machine, trucs et astuces sur ZX SPECTRUM.
SAGUEZ et ANDRIEUX	- Maîtrisez les interfaces de votre micro-ordinateur.
DE GEETER	- Forth pour micros.
ASTIER	- ATMOS à la conquête des jeux.
CROWTHER et HARTLEY	- MO5 et TO7 à la conquête des jeux.
MONTEIL	- Introduction à l'IBM-PC Junior.
OROS	- La conduite des ATARI XL - 600 XL, 800 XL, 1245 XLD.
PERBOST	- ZX SPECTRUM à la conquête des jeux.
BERNARD	- L'assembleur facile du 6809.
GARREC et VIGNET	- Des extensions à construire pour votre ORIC-ATMOS.

LOGILIVRES EYROLLES (logiciels sur cassettes)

PELLIER	- Kamikaze (jeu pour ZX Spectrum).
PELLIER	- Astéroïdes (jeu pour ZX Spectrum).
PELLIER	- Othello/Isola (jeux pour ZX Spectrum).
PELLIER	- Éditeur/Assembleur pour ZX Spectrum.
PERBOST et MASSE	- VIC 20 version de base à la conquête des jeux.
HADDADI	- Calcul des structures sur PC 1500/PC 2.
VANRYB et POLITIS	- LYNX. Dictée musicale. Générateur de caractères.
ROSENTHAL	- Résistance des matériaux sur ORIC-1 et ORIC-ATMOS.

Si vous désirez être tenu au courant de nos publications, il vous suffit d'adresser votre carte de visite au :

Service «Presse» Éditions EYROLLES
61, Boulevard Saint-Germain,
75240 PARIS CEDEX 05,

en précisant les domaines qui vous intéressent. Vous recevrez régulièrement un avis de parution des nouveautés en vente chez votre libraire habituel.

Table des matières

1. Techniques employées pour la programmation de jeux en Basic sur ZX Spectrum 1

1.1. *Déplacement d'un mobile* 2

1.1.1. Principe 2
1.1.2. Évaluation d'expressions logiques 3
1.1.3. Technique du PRINT OVER. Création de "sprites" ... 5

1.2. *Utilisation du clavier* 8

1.2.1. Instruction INKEY$ 8
1.2.2. Fonction IN 8

1.3. *Utilisation des manettes de jeu* 12

1.4. *Utilisation de quelques variables système* 15

1.4.1. Scrolling continu 15
1.4.2. Bip clavier 17
1.4.3. Auto-repeat 17
1.4.4. Accès à l'horloge interne 18

1.5. *Définition de caractères graphiques* 20

1.5.1. Principe 20
1.5.2. Programme d'aide à la définition de caractères graphiques : DESSIN 23

1.6. *Redéfinition du générateur de caractères* 34

1.6.1. Principe 35
1.6.2. Exemple 36

1.7. *Grahismes et sons* 37
1.7.1. Graphismes 37
1.7.2. Sons 44
1.8. *Routine de renumérotation* 45

2. Trucs et astuces pour la programmation de jeux en langage machine sur ZX Spectrum 47
2.1. *Généralités* 47
2.2. *Accès au langage machine* 48
2.3. *Chargement de code hexadécimal* 49
2.4. *Affichage de caractères sur l'écran* 50
2.5. *Effacement de l'écran* 52
2.6. *Scrolling de l'écran* 53
2.7. *Utilisation du générateur de sons* 53
2.8. *Redéfinition du générateur de caractères* 54

3. Jeux 57
3.1. *Alphabet* 57
3.2. *Chase* (version clavier) 63
3.3. *Chase* (version manettes de jeu) 71
3.4. *Chenille* 75
3.5. *Damier solitaire* 83
3.6. *Exploration Spatiale* (48 K) 89
3.7. *Kayak* 104
3.8. *Labyrinthe* 111
3.9. *Picman* 119
3.10. *Tarzan* 130

4. Adaptation des jeux pour ZX 81 139
4.1. *Différences entre ZX 81 et ZX Spectrum* 140
4.2. Exemple d'adaptation d'un jeu : *FIL D'ARIANE* 141

Annexes 148
1. L'écran du ZX Spectrum 148
2. Bibliothèque de caractères 150

1

Techniques employées pour la programmation de jeux en Basic sur ZX Spectrum

ZX Spectrum à la conquête des jeux a pour but non seulement de vous proposer de nombreux jeux plus passionnants les uns que les autres, mais aussi de vous faire découvrir les techniques les plus employées pour la programmation de jeux en Basic sur ZX Spectrum. Ce premier chapitre vous montre en particulier comment déplacer un mobile sur l'écran du téléviseur, enlever l'auto-repeat du clavier, utiliser l'horloge interne du ZX Spectrum, ... En utilisant les nombreux "trucs" que vous trouverez dans ces quelques pages, vous serez ainsi capable de programmer vous-même de très beaux jeux.

La lecture de ce chapitre présuppose connue la programmation en Basic. Pour les débutants, le livre "LE BASIC UNIVERSEL" publié dans cette même collection fournira toutes les bases nécessaires à une bonne compréhension.

1.1. DÉPLACEMENT D'UN MOBILE

1.1.1. Principe

Beaucoup de jeux utilisent les déplacements de différents mobiles : balle, raquette, missile, voiture, ... Il est donc important de savoir déplacer correctement un mobile sur l'écran du téléviseur.

Pour recréer l'illusion du mouvement, une méthode d'animation couramment utilisée consiste à afficher un caractère correspondant au mobile à un emplacement donné. Après un court laps de temps, un blanc (espace) est mis à cet emplacement pour effacer le caractère. Ce court laps de temps a généralement permis de calculer les nouvelles coordonnées du caractère représentant le mobile, soit automatiquement (balle qui rebondit contre un mur par exemple), soit à partir des données entrées par le joueur sur le clavier (déplacement du petit glouton dans le jeu PICMAN). On affiche alors le caractère à sa nouvelle position et on recommence ainsi de suite, image après image comme dans un dessin animé.

La vitesse de déplacement du mobile dépend évidemment de la distance entre deux positions successives du mobile. Cette distance ne doit pas être trop grande pour que le mouvement ne paraisse pas saccadé.

Il faut aussi que le caractère reste visible suffisamment longtemps pour éviter l'impression de clignotement de l'image. Cea dépend de l'intervalle de temps entre deux apparitions du mobile.

Exemple d'application :

```
  10 BORDER 0: PAPER 0: INK 4
  20 CLS
  30 INPUT "distance entre deux
apparitions successives ",pas: L
ET pas=INT (ABS pas)
  40 IF pas=0 THEN STOP
  50 INPUT "temps separant deux
apparitions successives ",duree
  60 FOR i=0 TO 31 STEP pas
  70 PRINT AT 10,i;"*": BEEP .01
,10
  80 FOR j=1 TO duree: NEXT j
  90 PRINT AT 10,i;" "
 100 NEXT i
 110 GO TO 20
```

Utilisez ce programme en prenant des valeurs comprises entre 1 et 5 pour PAS et 1 à 200 pour DUREE. Nous vous laissons le soin de trouver vous-même les paramètres qui vous semblent convenir le mieux.

REMARQUE : Les lignes 10 et 20 donnent à l'écran et à son cadre la couleur noire ; les caractères seront écrits en vert. Cette combinaison de couleurs est celle qui assure la plus grande longévité à l'écran de votre téléviseur et celle qui fatigue le moins votre vue.

Supposons maintenant qu'un joueur doive déplacer sur l'écran une petite étoile à l'aide des quatre flèches : → (touche 8), ← (touche 5), ↓ (touche 6) et ↑ (touche 7).

Le programme peut être le suivant :

```
  10 BORDER 0: PAPER 0: INK 4
  20 CLS
  30 INPUT "distance entre deux
apparitions successives ";pas: L
ET pas=INT (ABS pas)
  40 IF pas=0 THEN STOP
  50 LET x=0: LET y=0
  60 PRINT AT y,x;"*": BEEP 0.01
,5
  70 LET hx=x: LET hy=y
  80 LET a$=INKEY$: IF a$="" THE
N GO TO 80
  90 IF a$="8" THEN LET x=x+pas:
 GO TO 130
 100 IF a$="5" THEN LET x=x-pas:
 GO TO 130
 110 IF a$="6" THEN LET y=y+pas:
 GO TO 130
 120 IF a$="7" THEN LET y=y-pas
 130 PRINT AT hy,hx;" "
 140 GO TO 60
```

1.1.2. Évaluation d'expressions logiques

Vous avez certainement remarqué en utilisant le petit programme donné ci-dessus, que l'étoile ne se déplaçait pas à la même vitesse dans toutes les directions. De plus la méthode employée par ce programme (tests successifs et branchements inconditionnels) est lourde et peu rapide.

Il est possible d'améliorer grandement ce programme en utilisant une propriété intéressante du ZX Spectrum en matière d'évaluation d'expressions logiques.

Le ZX Spectrum donne en effet la valeur 1 à une expression logique "vraie" et la valeur Ø à une expression logique "fausse".

Essayez par exemple:

LET a=Ø [ENTER]
PRINT (a=Ø) [ENTER]

Le résultat est alors 1.

PRINT (a=3)

Le résultat est alors Ø.

De même:

PRINT (A>3) donne Ø
PRINT (A<2) donne 1

Cela permet donc d'écrire le programme suivant (seules les lignes 9Ø et 1ØØ du programme précédent sont modifiées tandis que les lignes 11Ø et 12Ø disparaissent).

```
  10 BORDER 0: PAPER 0: INK 4
  20 CLS
  30 INPUT "distance entre deux
apparitions successives ";pas: L
ET pas=INT (ABS pas)
  40 IF pas=0 THEN STOP
  50 LET x=0: LET y=0
  60 PRINT AT y,x;"*": BEEP 0.01
,5
  70 LET hx=x: LET hy=y
  80 LET a$=INKEY$: IF a$="" THE
N GO TO 80
  90 LET x=x+pas*(a$="8")-pas*(a
$="5")
 100 LET y=y+pas*(a$="6")-pas*(a
$="7")
 130 PRINT AT hy,hx;" "
 140 GO TO 60
```

Vous remarquerez que ce programme est sensiblement plus rapide que le précédent tout en assurant un déplacement de l'étoile à une vitesse constante quelque soit le sens du déplacement.

Vous remarquerez aussi que ce programme possède le défaut de permettre à l'étoile de franchir les bords de l'écran, ce qui provoque l'apparition d'un message d'erreur. Alors que l'élimination de ce défaut était difficile et se serait faite au détriment de la rapidité d'exécution dans le programme précédent, elle est ici très simple à faire.

Soit par exemple :

```
  10 BORDER 0: PAPER 0: INK 4
  20 CLS
  30 LET x=0: LET y=0
  40 PRINT AT y,x; OVER 1;"*": B
EEP 0.01,5
  50 LET hx=x: LET hy=y
  60 LET a$=INKEY$: IF a$="" THE
N GO TO 60
  70 LET x=x+(a$="8" AND x<31)-(
a$="5" AND x>0)
  80 LET y=y+(a$="6" AND y<21)-(
a$="7" AND y>0)
  90 PRINT AT hy,hx; OVER 1;"*"
 100 GO TO 40
```

Il vous sera désormais impossible de faire sortir l'étoile de l'écran.

1.1.3. Technique du PRINT OVER. Création de "Sprites"

• *L'instruction PRINT OVER 1*

Le Basic du ZX Spectrum offre aux programmeurs une instruction très puissante : OVER.

En effet OVER 1 permet de combiner par une opération de "Ou exclusif" deux caractères sur une même position de l'écran.

Essayez par exemple :

PRINT AT 1Ø,1Ø; "■"

un petit carré apparaît sur l'écran.

Tapez ensuite :

PRINT AT 1Ø,1Ø; OVER 1; "*"

vous voyez alors apparaître une étoile dans le carré précédemment affiché.

L'opération de "Ou exclusif" est involutive, c'est-à-dire que faire deux fois cette opération équivaut à ne rien faire.

Ainsi tapez à nouveau :

PRINT AT 10,10; OVER 1; "*"

la petite étoile disparaît et le carré redevient intact.

Cette méthode appliquée au déplacement d'un mobile permet d'écrire le programme suivant :

```
  10 BORDER 0: PAPER 0: INK 4
  20 CLS
  30 LET x=0: LET y=0
  40 PRINT AT y,x;"*": BEEP 0.01
,5
  50 LET hx=x: LET hy=y
  60 LET a$=INKEY$: IF a$="" THE
N GO TO 60
  70 LET x=x+(a$="6" AND x<31)-(
a$="5" AND x>0)
  80 LET y=y+(a$="6" AND y<21)-(
a$="7" AND y>0)
  90 PRINT AT hy,hx;" "
 100 GO TO 40
```

● *Création de "Sprites"*

La méthode donnée ci-dessus laisse entrevoir la possibilité d'obtenir sur le ZX Spectrum des "Sprites" comparables à ceux offerts par d'autres micro-ordinateurs.

Rappelons tout d'abord que les "Sprites" sont des caractères qui peuvent se déplacer comme des "fantômes" sur l'écran, c'est-à-dire sans altérer le contenu des positions de l'écran qui sont sur leur trajectoire.

La méthode classique de déplacement d'un mobile (affichage du mobile sur sa nouvelle position et effacement par un "blanc") présente l'inconvénient de détruire irrémédiablement le contenu des positions traversées par le mobile. Cela a pour effet de provoquer l'apparition d'une trainée dans la trajectoire du mobile et la destruction du décor dessiné sur l'écran. Or il est parfois indispensable de préserver certains dessins sur l'écran, même si un mobile est amené à passer dessus dans son déplacement. On peut citer par exemple le jeu PICMAN que vous trouverez dans ce livre : le glouton mange les vitamines dès qu'il passe

dessus (déplacement classique) alors que les monstres laissent ces mêmes vitamines intactes (déplacement à l'aide de PRINT OVER 1).

L'instruction PRINT OVER 1 permet d'obtenir un déplacement "transparent" d'un mobile (le mobile se superpose au décor qu'il traverse puis disparaît en laissant le décor intact grâce à la propriété d'involutivité du "ou exclusif"); le mobile se comporte alors comme un véritable "sprite".

Vous remarquerez toutefois que l'instruction PRINT OVER 1 ne préserve pas les attributs (couleur, clignotement, surbrillance) des positions traversées. Heureusement le ZX Spectrum dispose d'instructions complémentaires qui permettent d'obtenir une transparence totale au niveau des attributs:

```
PAPER 8
INK 8
FLASH 8
BRIGHT 8
```

Vous trouverez dans le programme donné ci-dessous un exemple de déplacement d'un "sprite" dans un environnement de carrés multicolores.

```
  10 BORDER 0: PAPER 0: INK 4
  20 CLS
  30 LET x=0: LET y=0: GO SUB 10
00
  40 PRINT AT y,x; OVER 1; INK 8
;"■": BEEP 0.01,5
  50 LET hx=x: LET hy=y
  60 LET a$=INKEY$: IF a$="" THE
N GO TO 60
  70 LET x=x+(a$="8" AND x<31)-(
a$="5" AND x>0)
  80 LET y=y+(a$="6" AND y<21)-(
a$="7" AND y>0)
  90 PRINT AT hy,hx; OVER 1; INK
 8;"■"
 100 GO TO 40
1000 FOR i=0 TO 7
1010 READ n
1020 POKE USR "■"+i,n
1030 NEXT i
1040 DATA 126,255,219,255,231,25
5,165,165
1050 FOR i=1 TO 10
1060 PRINT AT INT (RND*22),INT (
RND*31); INK RND*6+1;"■"
1070 NEXT i
1080 RETURN
```

REMARQUE : Le caractère graphique est obtenu en tapant la lettre A en mode graphique (lignes 4Ø, 9Ø et 1Ø2Ø).

1.2. UTILISATION DU CLAVIER

Le clavier est le premier périphérique d'entrée d'un micro-ordinateur. C'est par son intermédiaire qu'un joueur peut contrôler le déplacement d'un mobile, le tir d'un missile, etc... Il est donc très important de bien savoir utiliser le clavier du ZX Spectrum pour programmer des jeux.

1.2.1. Instruction INKEY$

L'instruction INKEY$ permet de lire "au vol" un caractère entré à partir du clavier alors que l'instruction INPUT exige la validation de l'entrée par pression de la touche ENTER. L'instruction INKEY$ est donc très souvent utilisée pour les jeux.

Les programmes, donnés précédemment pour le déplacement d'un mobile, sont de bons exemples d'utilisation de INKEY$.

1.2.2. Fonction IN

La fonction IN a pour résultat un nombre entier, compris entre Ø et 255, qui correspond à l'octet lu sur le port d'adresse *n*.

Le clavier du ZX Spectrum est divisé en huit demi-rangées de cinq touches. Chacune de ces demi-rangées est considérée comme un port d'entrée et peut donc, à ce titre, être lue avec la fonction IN.

Le tableau ci-dessous donne la correspondance entre le numéro de la demi-rangée, l'adresse du port d'entrée associé et les touches qui composent cette demi-rangée.

Numéro demi-rangée	*Adresse*	*Touches*
Ø	65278	CAPS SHIFT, Z, X, C, V
1	65Ø22	A, S, D, F, G
2	6451Ø	Q, W, E, R, T
3	63486	1, 2, 3, 4, 5
4	61438	6, 7, 8, 9, Ø
5	57342	Y, U, I, O, P
6	4915Ø	H, J, K, L, ENTER
7	32766	B, N, M, SYMBOL SHIFT, SPACE

La formule qui permet d'obtenir l'adresse (@) du port correspondant à une demi-rangée et le numéro (*n*) de cette demi rangée est:

@ = 254 + 256 * (255 − 2↑n)

Exemple: Demi-rangée n° 4:

@ = 254 + 256 * (255 − 2 4)
@ = 61438

L'octet lu par la fonction IN est de la forme:

b_7	b_6	b_5	b_4	b_3	b_2	b_1	b_0
1	1	1	X	X	X	X	X

Les bits b_0, b_1, b_2, b_3 et b_4 correspondent aux cinq touches de la demi-rangée, le bit b_0 pour la touche la plus proche du centre du clavier et le bit b_4 pour la touche extérieure.

Exemple :

b_7	b_6	b_5	b_4	b_3	b_2	b_1	b_0
1	1	1	Q	W	E	R	T

b_7	b_6	b_5	b_4	b_3	b_2	b_1	b_0
1	1	1	P	O	I	U	Y

Le bit qui représente une touche est à Ø si la touche est pressée alors qu'il reste à 1 si la touche est au repos.

Exemples :

Touche enfoncée	*Octet lu*	*Code obtenu*
Q	1 1 1 1 1 1 1 Ø	128 + 64 + 32 + 16 + 8 + 4 + 2 = 254
W	1 1 1 1 1 1 Ø 1	128 + 64 + 32 + 16 + 8 + 4 + 1 = 253
E	1 1 1 1 1 Ø 1 1	128 + 64 + 32 + 16 + 8 + 2 + 1 = 251
R	1 1 1 1 Ø 1 1 1	128 + 64 + 32 + 16 + 4 + 2 + 1 = 247
T	1 1 1 Ø 1 1 1 1	128 + 64 + 32 + 8 + 4 + 2 + 1 = 239
Y	1 1 1 1 1 1 1 Ø	128 + 64 + 32 + 16 + 8 + 4 + 2 = 254
U	1 1 1 1 1 1 Ø 1	128 + 64 + 32 + 16 + 8 + 4 + 1 = 253
I	1 1 1 1 1 Ø 1 1	128 + 64 + 32 + 16 + 8 + 2 + 1 = 251
O	1 1 1 1 Ø 1 1 1	128 + 64 + 32 + 16 + 4 + 2 + 1 = 247
P	1 1 1 Ø 1 1 1 1	128 + 64 + 32 + 8 + 4 + 2 + 1 = 239
Touche enfoncée	*Octet lu*	*Code obtenu*

REMARQUES :

- si aucune touche de la demi-rangée scrutée n'est pressée, c'est le code 255 qui est obtenu (octet lu : 11111111),
- si deux touches (ou plus) d'une même demi-rangée sont pressées simultanément, le code obtenu est calculé comme précédemment.

Exemple : Touches Q et W enfoncées en même temps :

octet lu : 11111ØØ

code obtenu : 128 + 64 + 32 + 16 + 8 + 4 = 252

En pratique le nombre de combinaisons ainsi produites conduit à interdire au joueur d'appuyer sur plusieurs touches à la fois afin de ne pas compromettre la rapidité du jeu par de nombreux tests.

Exemple d'utilisation :

```
  10 BORDER 0: PAPER 0: INK 4
  20 CLS
  30 LET x=0: LET y=0
  40 PRINT AT y,x;"*": BEEP 0.01
,5
  50 LET hx=x: LET hy=y
  60 LET a=IN 64510: IF a=255 TH
EN GO TO 60
  70 LET x=x+(a=247 AND x<31)-(a
=254 AND x>0)
  80 LET y=y+(a=253 AND y<21)-(a
=251 AND y>0)
  90 PRINT AT hy,hx;" "
 100 GO TO 40
```

Les touches qui permettent de déplacer la petite étoile sont :

Q (←)
W (↓)
E (↑)
R (→)

Lire le clavier avec la fonction IN est totalement inutile dans le cas de jeux à un seul joueur, et ce d'autant plus que les touches 5, 6, 7, 8, qui sont souvent les plus employées, n'appartiennent pas à une même demi-rangée. Cette façon de procéder est par contre indispensable pour programmer des jeux à deux joueurs. En effet l'instruction INKEY$ ne

permet de lire qu'une seule touche. Si l'un des joueurs appuie en permanence sur le clavier, l'autre joueur ne peut plus jouer. La fonction IN ne présente pas cet inconvénient. Il suffit d'affecter à chacun des joueurs une demi-rangée de touches et de lire avec la fonction IN ces demi-rangées.

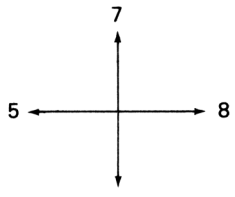
Vous trouverez un exemple complet dans le programme CHASE (version clavier).

1.3. UTILISATION DES MANETTES DE JEU

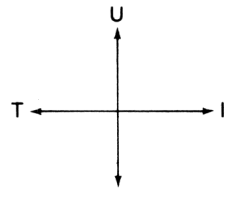
Jouer à partir du clavier n'est pas toujours facile et agréable, surtout dans le cas de jeux à deux joueurs. Le ZX Spectrum permet le raccordement de manettes de jeu par l'intermédiaire d'une interface spéciale qui se branche sur le bus d'extension.

REMARQUE IMPORTANTE : De même que l'imprimante, l'interface pour manettes de jeu ne peut être mise sur le bus d'extension que si l'alimentation du ZX Spectrum est *éteinte.*

L'interface a été conçue de façon à simuler les touches 5, 6, 7, 8 et Ø pour la manette 1 et les touches T, Y, U, I et P pour la manette 2.



FEU = Ø
manette 1



FEU = P
manette 2

Tous les programmes scrutant les touches 5, 6, 7 et 8 à l'aide de l'instruction INKEY$ (c'est le cas de la majorité des jeux de ce livre) peuvent donc fonctionner sans aucune modification avec la manette de jeu 1.

Essayez en particulier les programmes donnés dans la partie 1.1 de ce chapitre pour déplacer la petite étoile avec la manette de jeu 1.

Le petit programme donné ci-dessous permet, en plus du déplacement de l'étoile, l'utilisation du bouton FEU de la manette de jeu 1.

```
  10 BORDER 0: PAPER 0: INK 4
  20 CLS
  30 LET x=10: LET y=5
  35 OUT 63486,248
  40 PRINT AT y,x;"*"
  50 LET hx=x: LET hy=y
  60 LET a=255-IN 63486: IF a=16
THEN LET a=32
  66 LET a=a+(255-IN 61438)
  67 LET feu1=(a/2<>INT (a/2))
  69 LET a=a-feu1
  70 LET x=x+(a=4 AND x<20)-(a=3
2 AND x>10)
  80 LET y=y+(a=16 AND y<15)-(a=
8 AND y>5)
  85 IF feu1=1 THEN GO SUB 500
  90 PRINT AT hy,hx;" "
 100 GO TO 40
 500 PRINT AT y,x; FLASH 1;"*"
 510 LET px=8*x+4: LET py=8*(21-
y)+4
 520 FOR j=1 TO 2
 530 FOR i=1 TO 7
 540 BEEP 0.01,10
 550 CIRCLE INK i, OVER 1;px,py,
i*6
 560 NEXT i
 570 NEXT j
 580 RETURN
```

Il semblerait à priori possible de programmer un jeu à deux joueurs qui utilisent tous les deux les manettes de jeu. Il suffit en effet de tester les touches 5, 6, 7, 8 et Ø pour la partie de jeu concernant le premier joueur et les touches T, Y, U, I et P pour la partie concernant le deuxième joueur. Malheureusement, comme nous l'avons déjà dit précédemment, la fonction INKEY$ ne permet de lire qu'une touche à la fois et par là même une seule manette de jeu à la fois. Si un joueur maintient sa manette dans une position donnée, l'autre joueur est bloqué.

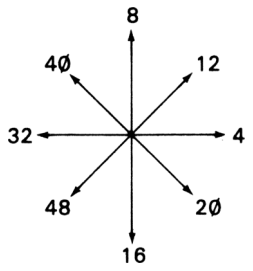
Pour remédier à cet inconvénient, il faut utiliser la fonction IN en prenant toutefois garde au fait que les touches 5, 6, 7, 8 et Ø n'appartiennent pas à une même demi-rangée (même problème pour les touches T, Y, U, I et P).

Cela nous conduit à écrire les programmes de lecture des manettes de jeu de la façon suivante :

lecture manette 1 : LET a=255–IN 63486 : IF a=16 THEN LET a=32
LET a=a+(255–IN 61438)

lecture manette 2 : LET b=255–IN 64510 : IF b=16 THEN LET b=32
LET b=b+(255–IN 57342)

a et b prennent alors les valeurs :



Si la manette n'est pas manœuvrée, le code obtenu est 0.

Si le bouton FEU est pressé, il faut ajouter 1 aux codes précédents. Afin d'éviter de trop nombreux tests qui nuiraient à la rapidité du jeu, il est intéressant d'isoler l'information de mise à feu de la façon suivante :

manette 1 : LET feu1=(a/2 <> INT (a/2))
LET a=a–feu1

manette 2 : LET feu2=(b/2 <> INT (b/2))
LET b=b–feu2

Exemple :

```
  10 BORDER 0: PAPER 0: INK 4
  20 CLS
  30 LET x=10: LET y=5
  40 PRINT AT y,x;"*"
  50 LET hx=x: LET hy=y
  60 LET a$=INKEY$: IF a$="" THE
N GO TO 60
  70 LET x=x+(a$="8" AND x<20)-(
a$="5" AND x>10)
  80 LET y=y+(a$="6" AND y<15)-(
a$="7" AND y>5)
  85 IF a$="0" THEN GO SUB 500
  90 PRINT AT hy,hx;" "
```

```
 100 GO TO 40
 500 PRINT AT y,x; FLASH 1;"*"
 510 LET px=8*x+4: LET py=8*(21-
y)+4
 520 FOR j=1 TO 2
 530 FOR i=1 TO 7
 540 BEEP 0.01,10
 550 CIRCLE INK i; OVER 1;px,py,
i*6
 560 NEXT i
 570 NEXT j
 580 RETURN
```

Utilisez la manette 1 pour déplacer l'étoile.

Vous trouverez un exemple complet dans le programme CHASE (version joystick).

1.4. UTILISATION DE QUELQUES VARIABLES SYSTÈME

Le ZX Spectrum réserve en RAM un espace compris entre les adresses 23552 et 23734 pour des variables qui sont utilisées par le système d'exploitation (d'où le nom de variables "système").

Certaines de ces variables peuvent être modifiées par le programmeur pour obtenir des effets spéciaux ou des fonctions particulières. Les paragraphes suivants ont pour but de vous montrer comment agir sur quelques variables système pour changer le clic du clavier, enlever l'auto-repeat du clavier, accéder à l'horloge interne, ...

1.4.1. Scrolling continu

Dès que l'écran du téléviseur est plein, le ZX Spectrum interrompt l'exécution ou la liste de son programme et pose la question SCROLL? Si l'utilisateur presse soit la touche N soit la touche SPACE, l'exécution est interrompue alors que la pression de toute autre touche fait reprendre le défilement jusqu'à ce que l'écran soit à nouveau plein.

Essayez par exemple:

```
10 FOR i=1 TO 100
20 PRINT i
30 NEXT i
```

Cette possibilité est particulièrement intéressante lorsqu'on consulte le listing d'un long programme mais peut s'avérer très ennuyeuse pour certains jeux qui nécessitent un défilement continu de l'écran.

Il est fort heureusement possible d'agir sur une variable système du ZX Spectrum pour obtenir ce défilement continu. Cette variable a pour adresse : 23692

A l'initialisation, le système place à cette adresse la valeur 1.

Faites :

```
PRINT PEEK 23692
```

pour vous en convaincre.

Le défilement continu s'obtient en mettant à l'adresse 23692 un nombre supérieur à 2, par exemple 255.

Essayez maintenant :

```
10 FOR i=1 TO 100
20 PRINT i
30 POKE 23692,255
40 NEXT i
```

Le petit programme de jeu, donné ci-dessous, vous montre comment utiliser cette astuce pour simuler la descente d'un canöe kayak dans un torrent parsemé de rochers dangereux.

```
  10 BORDER 1: PAPER 1: INK 4: C
LS
  20 LET x=15
  30 POKE 23692,255
  40 PRINT AT 21,INT (RND*31);"*
** ***": PRINT
  50 LET a$=INKEY$
  60 LET x=x+(a$="6" AND x<31)-(
a$="5" AND x>0)
  70 IF SCREEN$ (1,x)="*" THEN G
O TO 100
  80 PRINT AT 0,x; INK 7;"U"
  90 GO TO 30
 100 PRINT AT 1,x; INK 2; FLASH
1;"U"
 110 PRINT AT 21,0;"CRAAAAASH !!
!!!"
 120 PRINT "glou glou..."
```

Utilisez les touches 5 et 8 pour déplacer le canöe. Le jeu sera fini dès que vous heurterez un rocher.

La ligne 4Ø permet d'obtenir un scrolling continu de l'écran, créant ainsi une animation qui donne l'illusion du déplacement du canöe dans le torrent.

Vous trouverez une version très améliorée de ce jeu dans le programme KAYAK.

1.4.2. Bip clavier

Le clic du clavier est déterminé par la valeur de la variable système d'adresse 236Ø9.

A l'initialisation :

```
PEEK 236Ø9=Ø
```

Vous pouvez changer cette valeur pour obtenir un son plus audible chaque fois qu'une touche est pressée, ce qui est particulièrement utile lorsqu'on doit taper de longs programmes.

Essayez par exemple :

```
POKE 236Ø9,1ØØ
```

1.4.3. Auto-repeat

Le clavier du ZX Spectrum est doté d'un dispositif d'auto-repeat automatique dès qu'une touche est pressée plus d'un certain temps.

La durée pendant laquelle une touche doit être maintenue enfoncée pour qu'il n'y ait pas répétition est déterminée par les variables système d'adresses 23561 et 23562.

Initialement :

```
PEEK 23561=35
PEEK 23562=5
```

Pour enlever l'auto-repeat, il suffit de faire :

```
POKE 23561,255
POKE 23562,255
```

Cette astuce est parfois utilisée dans certains jeux où seule la rapidité du joueur pour presser les touches doit intervenir.

Essayez de placer différentes valeurs aux adresses 23561 et 23562 ; votre clavier aura alors un comportement parfois surprenant.

1.4.4. Accès à l'horloge interne

Le ZX Spectrum possède une horloge interne à laquelle le programmeur peut accéder par les variables systèmes d'adresses 23672, 23673 et 23674.

La formule 65536 * PEEK 23674 + 256 * PEEK 23673 + PEEK 23672 donne le nombre de cinquantièmes de seconde écoulés depuis la mise sous tension de l'ordinateur (ou depuis la dernière mise à zéro de l'horloge).

Pour initialiser l'horloge interne du ZX Spectrum à une valeur donnée, il faut d'abord convertir cette valeur en cinquantièmes de seconde puis "décomposer" le nombre ainsi obtenu en trois octets.

Exemple: L'heure d'initialisation doit être 15 h 3Ø. Cela représente (15 * 6Ø + 3Ø) * 6Ø * 5Ø cinquantièmes de seconde, soit 279ØØØØ cinquantièmes de seconde.

279ØØØØ=65536*42+256*146+112

Pour mettre l'horloge à l'heure, il faut donc faire :

```
POKE 23674,42
POKE 23673,146
POKE 23672,112
```

L'horloge interne du ZX Spectrum peut être très utile dans certains jeux pour chronométrer le joueur (temps mis pour sortir d'un labyrinthe, temps accordé pour jouer dans un jeu de réflexion).

Attention, les instructions BEEP, SAVE, LOAD entraînent un "arrêt" du chronomètre. Une mesure précise de temps et la génération de sons dans un jeu sont donc généralement incompatibles.

Le programme de jeu REFLEXE donné ci-dessous met en œuvre l'horloge interne du ZX Spectrum pour compter le temps que vous

mettrez pour trouver la touche correspondant au caractère affiché sur l'écran.

```
  10 BORDER 0: PAPER 0: INK 4: C
LS
  20 DEF FN t()=INT ((65536*PEEK
 23674+256*PEEK 23673+PEEK 23672
)*0.2)
  30 LET i=INT (RND*89+33): IF i
=91 OR i=92 OR i=93 THEN GO TO 3
0
  40 LET a$=CHR$ i
  50 LET x=INT (RND*32): LET y=I
NT (RND*20)
  60 FOR i=0 TO 2
  70 POKE 23672+i,0
  80 NEXT i
  90 PRINT AT y,x;a$
 100 LET b$=INKEY$: IF b$<>a$ TH
EN GO TO 100
 110 PRINT AT 21,0;"TEMPS : ";FN
 t();" dixiemes de seconde"
 130 PAUSE 0
 140 PAUSE 0
 150 RUN
```

La structure de ce petit programme est la suivante:

Ligne 1Ø: l'écran et le bord de l'écran sont de couleur noire alors que l'encre des caractères sera verte. Ce choix de couleurs est ainsi fait pour ménager à la fois votre téléviseur et votre vue.

Ligne 2Ø: définition d'une fonction qui donne le nombre de dixièmes de seconde écoulés depuis la dernière mise à zéro de l'horloge.

Lignes 3Ø et 4Ø: choix d'un caractère avec élimination de certains caractères accessibles seulement en mode étendu.

Ligne 5Ø: choix d'une position sur l'écran.

Lignes 6Ø à 8Ø: mise à zéro de l'horloge.

Ligne 9Ø: affichage du caractère choisi.

Ligne 1ØØ: cette ligne boucle sur elle-même jusqu'à ce que le joueur ait pressé la bonne touche.

Ligne 11Ø: affichage du nombre de dixièmes de seconde écoulés entre le moment où le caractère est apparu sur l'écran et l'instant où le joueur a réussi à presser la touche correspondante.

Lignes 13Ø et 14Ø: l'exécution du programme s'interrompt jusqu'à ce que le joueur presse une touche quelconque. L'instruction PAUSE Ø est dupliquée pour éviter qu'un rebond du clavier, ou une pression trop prolongée d'une touche par le joueur, ne provoque un relancement trop rapide du jeu.

Ligne 15Ø: relancement du jeu.

1.5. DÉFINITION DE CARACTÈRES GRAPHIQUES

1.5.1. Principe

● *Caractères semi-graphiques*

Le générateur de caractères du ZX Spectrum permet d'afficher, en plus des caractères alphanumériques habituels, un ensemble de caractères semi-graphiques donnés ci-contre (codes ASCII compris entre 128 et 136).

Ces caractères semi-graphiques sont obtenus avec les touches 1, 2, ..., 8 en mode graphique G. Leur emploi est peu pratique et donc peu fréquent. Il est bien préférable d'utiliser la possibilité, offerte par le ZX Spectrum, de redéfinir à volonté certains caractères.

● *Caractères graphiques définis par l'utilisateur*

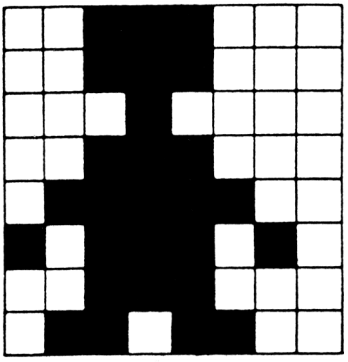
Il est possible de redéfinir très simplement vingt et un caractères (de codes ASCII compris entre 144 et 164) à l'aide des fonctions POKE, USR et BIN.

Le codage se fait sur une matrice 8 × 8.

Symbole	*Code*	*Touches*	*Symbole*	*Code*	*Touches*
	128	G 8		143	G shifted 8
	129	G 1		142	G shifted 1
	130	G 2		141	G shifted 2
	131	G 3		140	G shifted 3
	132	G 4		139	G shifted 4
	133	G 5		138	G shifted 5
	134	G 6		137	G shifted 6
	135	G 7		136	G shifted 7

Illustration extraite du manuel Sinclair.

Exemple:



0 0 1 1 1 0 0 0
0 0 1 1 1 0 0 0
0 0 0 1 0 0 0 0
0 0 1 1 1 0 0 0
0 1 1 1 1 1 0 0
1 0 1 1 1 0 1 0
0 0 1 1 1 0 0 0
0 1 1 0 1 1 0 0

La définition de ce petit bonhomme peut se faire de la façon suivante :

```
  10 BORDER 0: PAPER 0: INK 4: C
LS
  20 LET a$=CHR$ (144)
  30 POKE USR a$+0,BIN 00111000
  40 POKE USR a$+1,BIN 00111000
  50 POKE USR a$+2,BIN 00010000
  60 POKE USR a$+3,BIN 00111000
  70 POKE USR a$+4,BIN 01111100
  80 POKE USR a$+5,BIN 10111010
  90 POKE USR a$+6,BIN 00111000
 100 POKE USR a$+6,BIN 01101100
 110 PRINT AT 10,10;a$
```

Désormais vous obtiendrez l'affichage du petit bonhomme en passant en mode graphique G et en pressant la touche A. Même un NEW ne modifiera pas le nouveau caractère. Seule une coupure d'alimentation et une remise sous tension peuvent redonner au caractère sa configuration initiale (a).

L'emploi de la fonction BIN se justifie par le fait que l'on obtient très facilement les octets nécessaires pour la définition du caractère en mettant des 1 dans les cases noires et des Ø dans les cases blanches. Il est toutefois possible de calculer les valeurs décimales correspondantes et de "POKer" directement ces valeurs.

Le Basic du ZX Spectrum dispose des instructions READ et DATA, ce qui est très utile pour la définition de caractères graphiques.

Coupez l'alimentation de votre ordinateur puis remettez-le sous tension avant d'introduire le programme suivant :

```
  10 BORDER 0: PAPER 0: INK 4: C
LS
  20 LET a$=CHR$ (144)
  30 FOR i=0 TO 7
  40 READ a
  50 POKE USR a$+i,a
  60 NEXT i
  70 DATA 56,56,16,56,124,186,56
,108
  80 PRINT AT 10,10;a$
```

La possibilité de redéfinir ainsi juqu'à vingt et un caractères est particulièrement intéressante pour les jeux. La quasi-totalité des jeux de "ZX Spectrum à la conquête des jeux" fait appel à des caractères défi-

nis. Vous trouverez en annexe une bibliothèque des principaux caractères utilisés dans ce livre.

1.5.2. Programme d'aide à la définition de caractères graphiques : DESSIN

Pour définir des caractères graphiques, on est souvent obligé de trouver une feuille de papier, un crayon et une gomme, de dessiner une grille 8 × 8 et de faire de nombreux essais avant d'arriver enfin aux caractères souhaités. Afin d'éviter ce travail long et fastidieux, vous pourrez utiliser le programme d'aide à la définition de caractères graphiques qui est donné ci-après.

Avant de donner le listing du programme, étudions tout d'abord les principes qui ont présidé à son élaboration ainsi que les fonctions qu'il met à la disposition de son utilisateur.

Le programme DESSIN a été conçu de façon très structurée et très modulaire afin de faciliter son écriture, la mise au point et surtout afin de permettre toute évolution : ajout de nouvelles commandes, modification des commandes existantes, ...

L'utilisation du programme est très simple puisqu'il vous suffira de suivre les menus qui apparaissent sur l'écran pour vous guider dans le choix des fonctions disponibles.

- *Menu principal*

Le menu principal est le suivant :

```
>COM: Init Def Voir Jeux Fin
```

Passons en revue les différentes commandes proposées par ce menu.

- *Commande DEFINITION*

En tapant d ou D lorsque le menu principal est affiché, vous avez accès à la commande DEFINITION et le menu suivant apparaît :

```
>DEF: Chiff Dess Fin
```

— F(in

Presser f ou F permet de remonter au menu principal :

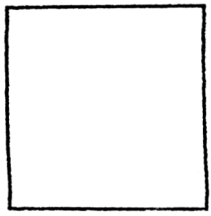
— C(hiff

En tapant c ou C, vous passez en mode de définition d'un caractère graphique par chiffres.

Le programme vous demande quel caractère vous voulez définir. Vous avez le choix entre A, B, C et D.

L'écran se présente ensuite de la façon suivante :

```
CHIFFRE
```

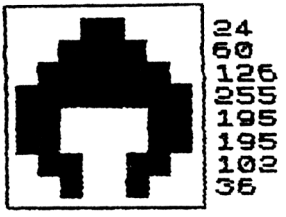


```
Donnee 1
```

Vous devez alors entrer les huit entiers qui correspondent aux huit lignes du tableau représentant le caractère. Dès que ces huit nombres ont été acceptés par la machine (tout entier supérieur à 255 est refusé), le caractère est affiché et le programme vous propose une copie sur imprimante.

Exemple :

```
CHIFFRE
```



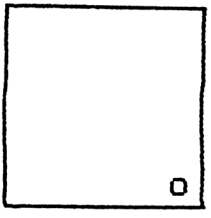
CARACTERE GRAPHIQUE 'C'

Cette façon de définir un caractère est particulièrement pratique quand on connaît déjà les DATA qui lui correspondent et que l'on désire simplement visualiser le caractère.

— D(ess

Le programme vous demande de choisir le caractère qui va être dessiné. Cela fait, l'écran prend la forme suivante :

```
>DES: N(oir E(ff V(al C(opy F(in
```



CARACTERE GRAPHIQUE 'A'

Sur l'exemple c'est le caractère A qui sera défini. Le O symbolise le curseur.

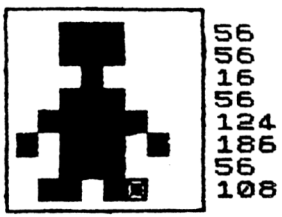
Vous pouvez déplacer ce O à l'aide des quatres touches 5 (←), 6 (↓), 7 (↑) et 8 (→). La manette de jeu 1 permet donc elle aussi de déplacer le O.

En appuyant sur n ou N, vous noircissez la case occupée par le O.

En appuyant sur e ou E, vous effacez (s'il y a lieu de le faire) la case occupée par le O.

Quand vous estimez que votre caractère est bien défini, vous pouvez le valider en pressant v ou V.

```
VALIDE
```




```
CARACTERE GRAPHIQUE 'A'
```

La commande V(al n'a pas un caractère définitif et vous pouvez encore modifier le caractère avant de le valider à nouveau. Par contre, si vous désirez conserver en mémoire le caractère défini, vous devez obligatoirement utiliser la commande V(al avant de faire F(in pour revenir au menu de niveau supérieur.

La commande C(opy permet de recopier sur l'imprimante tout ce qui est affiché sur l'écran.

- *Commande VOIR*

En tapant v ou V lorsque le menu principal est affiché, vous avez accès à la commande VOIR.

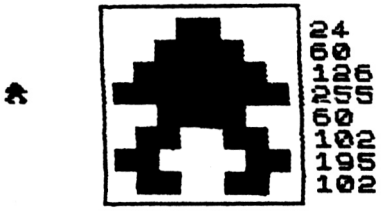
Vous devez préciser le caractère que vous désirez voir afficher. Ce caractère apparaît alors en petit et en grand avec la liste des entiers qui

permettent de le définir. Une copie sur imprimante vous est proposée. Le retour au menu principal est automatique.

Rappelons que le programme DESSIN n'est prévu que pour les quatre caractères A, B, C et D mais que vous pourrez, au prix de petites modifications, permettre la définition d'autres caractères (à concurrence de vingt et un toutefois).

Exemple :

```
VOIR
```



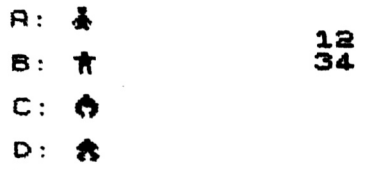
CARACTERE GRAPHIQUE 'D'

• *Commande J(ux*

En tapant j ou J lorsque le menu principal est affiché, vous avez accès à la commande JUXTAPOSITION.

L'écran se présente alors de la façon suivante :

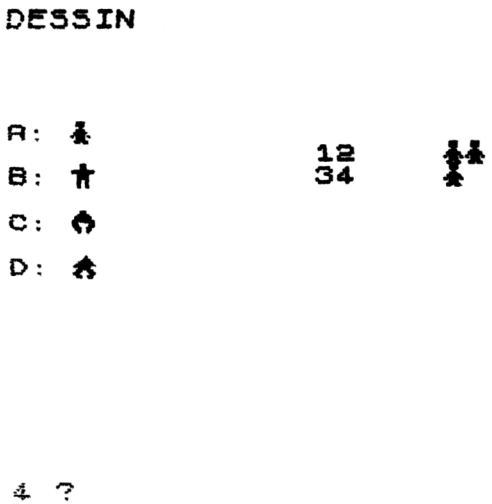
```
>JUX: D(ess I(mp F(in
```



Les quatre caractères A, B, C et D sont affichés. Ceux d'entre eux qui n'ont pas été définis au préalable sont représentés par un espace.

La fonction D(ess permet de juxtaposer dans une matrice 2 x 2 les caractères que vous avez défini.

Exemple :



Sur cet exemple le caractère graphique 'A' a déjà été choisi trois fois. Le programme attend le choix d'un quatrième caractère.

Il est ainsi possible de définir des macro-caractères, ce qui est parfois utile pour faire des dessins ou des jeux.

F(in vous permet de remonter au menu principal.

- *Commande INITIALISATION*

En tapant i ou I lorsque le menu principal est affiché, vous avez accès à la commande INITIALISATION.

le programme vous demande de préciser le caractère qui va être réinitialisé (c'est-à-dire transformé en "espace").

Pour revenir au menu principal, répondre par un mot commençant par une autre lettre que o ou O à la question "autre caractère (O/N)?".

- *Commande FIN*

La commande F(in du menu principal permet de sortir du programme.

Expliquer le programme DESSIN ligne à ligne est inutile. Sa structure très modulaire le rend très simple et très facile à comprendre. Vous remarquerez que tous les sous-programmes portent des noms et qu'ils sont toujours appelés en faisant référence à leurs noms. A ces noms sont associées des variables qui sont initialisées dès le lancement du programme. Vous pourrez donc très simplement déplacer, enlever, ajouter des modules.

```
  1 REM **********
  2 REM * DESSIN *
  3 REM **********
 10 REM
 20 REM PRESENTATION
 30 REM
 40 GO SUB 9000
 50 REM
 60 REM MODE COMMANDE
 70 REM
 80 CLS : PRINT AT 0,0;c$
 90 GO SUB saisie
100 IF a$="I" THEN GO SUB init:
GO TO 60
110 IF a$="D" THEN GO SUB defin
ition: GO TO 60
120 IF a$="V" THEN GO SUB voir:
GO TO 60
130 IF a$="J" THEN GO SUB jux:
GO TO 60
140 IF a$="F" THEN GO TO 180
150 GO SUB impossible
160 GO TO 60
170 CLS
180 STOP
999 REM
1000 REM INIT, remise a zero
1001 REM
1010 CLS : PRINT AT 0,0;"INITIAL
ISATION"
1020 GO SUB choix
1030 FOR i=0 TO 7
1040 POKE USR b$+i,0
1045 LET v(n*8+i+1)=0
1050 NEXT i
1060 INPUT "autre caractere (O/n
)",r$: IF r$="" THEN GO TO 1060
1070 IF r$(1)="o" OR r$(1)="O" T
HEN GO TO 1020
1080 RETURN
1999 REM
2000 REM DEFINITION
2001 REM
2010 CLS
2020 PRINT AT 0,0;d$
2030 GO SUB saisie
2040 IF a$="C" THEN GO SUB chiff
re: GO TO 2010
```

```
2050 IF a$="D" THEN GO SUB dessi
n: GO TO 2010
2060 IF a$="F" THEN GO TO 2090
2070 GO SUB impossible
2080 GO TO 2030
2090 RETURN
2999 REM
3000 REM VOIR
3001 REM
3010 CLS : PRINT AT 0,0;"VOIR"
3020 GO SUB choix
3022 FOR i=0 TO 7
3023 POKE USR b$+i,v(n*8+i+1)
3024 NEXT i
3025 PRINT AT 7,7;CHR$ (144+n)
3030 GO SUB affdata
3035 GO SUB affichage
3040 INPUT "Copie sur imprimante
 (o/n)",r$: IF r$="" THEN GO TO
3040
3050 IF r$(1)="o" OR r$(1)="O" T
HEN COPY
3060 RETURN
3999 REM
4000 REM JUXTAPOSITION
4001 REM
4010 CLS
4012 FOR j=0 TO 7
4013 POKE USR "A"+j,v(j+1)
4014 POKE USR "B"+j,v(9+j)
4015 POKE USR "C"+j,v(17+j)
4016 POKE USR "D"+j,v(25+j)
4017 NEXT j
4020 PRINT AT 0,0;f$
4030 PRINT AT 5,0;"A: ";CHR$ (14
4)
4040 PRINT AT 7,0;"B: ";CHR$ (14
5)
4050 PRINT AT 9,0;"C: ";CHR$ (14
6)
4060 PRINT AT 11,0;"D: ";CHR$ (1
47)
4070 PRINT AT 6,14;"12"
4080 PRINT AT 7,14;"34"
4090 GO SUB saisie
4100 IF a$="I" THEN COPY : GO TO
 4090
4110 IF a$="F" THEN GO TO 4300
4120 IF a$="D" THEN GO TO 4150
4130 GO SUB impossible
4140 GO TO 4090
4150 PRINT AT 0,0;"
          ";AT 0,0;"DESSIN"
4155 FOR i=1 TO 4
4160 PRINT AT 21,0;i;" ?"
4170 GO SUB saisie
4180 LET x=20+(i=2)+(i=4)
4190 LET y=6+(i=3)+(i=4)
```

```
4200 IF CODE a$>63 AND CODE a$<6
9 THEN PRINT AT y,x;CHR$ (CODE a
$+79): GO TO 4270
4240 GO SUB impossible
4250 GO TO 4160
4270 NEXT i
4280 PRINT AT 21,0;"    "
4290 GO TO 4020
4300 RETURN
4999 REM
5000 REM CHIFFRE
5001 REM
5005 CLS : PRINT AT 0,0;"CHIFFRE
5010 GO SUB choix
5020 GO SUB affinit
5030 FOR i=1 TO 8
5040 PRINT AT 21,0;"Donnee ";i:
INPUT h
5050 LET h=INT (ABS h)
5060 IF h>255 THEN BEEP 0.1,10:
GO TO 5040
5070 LET v(n*8+i)=h
5080 NEXT i
5085 PRINT AT 21,0;"          "
5090 GO SUB affichage
5095 GO SUB affdata
5097 PRINT AT 7,7;CHR$ (144+n)
5100 INPUT "Copie sur imprimante
 (o/n)",r$: IF r$="" THEN GO TO
5100
5110 IF r$(1)="o" OR r$(1)="O" T
HEN COPY
5120 RETURN
5999 REM
6000 REM DESSIN
6001 REM
6010 CLS
6015 PRINT AT 0,0;"DESSIN
    "
6020 GO SUB choix
6025 GO SUB affichage
6030 PRINT AT y+3,x+11; OVER 1;"
O"
6040 PRINT AT 0,0;e$
6050 GO SUB saisie
6060 IF a$="N" THEN GO SUB noir:
 GO TO 6050
6070 IF a$="E" THEN GO SUB effac
e: GO TO 6050
6080 IF a$="V" THEN GO SUB valid
e: GO TO 6040
6090 IF a$="C" THEN COPY : GO TO
 6050
6100 IF a$="F" THEN GO TO 6200
6110 IF a$="8" AND x<8 THEN LET
x=x+1: GO TO 6160
6120 IF a$="5" AND x>1 THEN LET
x=x-1: GO TO 6160
```

```
6130 IF a$="7" AND y>1 THEN LET
y=y-1: GO TO 6160
6140 IF a$="6" AND y<8 THEN LET
y=y+1: GO TO 6160
6150 GO SUB impossible: GO TO 60
50
6160 PRINT AT hy+3,hx+11; OVER 1
;"O"
6170 LET hy=y: LET hx=x
6180 PRINT AT y+3,x+11; OVER 1;"
O"
6190 GO TO 6050
6200 RETURN
6299 REM
6300 REM NOIR
6301 REM
6310 PRINT AT y+3,x+11;"■";AT y+
3,x+11; OVER 1;"O"
6320 LET t(y,x)=1
6330 RETURN
6399 REM
6400 REM EFFACE
6401 REM
6410 PRINT AT y+3,x+11;" ";AT y+
3,x+11; OVER 1;"O"
6420 LET t(y,x)=0
6430 RETURN
6499 REM
6500 REM VALIDE
6501 REM
6505 PRINT AT 0,0;"VALIDE
                "
6510 FOR j=1 TO 8
6520 LET a=0
6530 FOR i=1 TO 8
6540 IF t(j,i)=1 THEN LET a=a+2↑
(8-i)
6550 NEXT i
6560 LET v(n*8+j)=a
6570 PRINT AT j+3,21;a
6580 NEXT j
6590 RETURN
6999 REM
7000 REM AFFICHAGE
7001 REM
7025 FOR i=1 TO 8: FOR j=1 TO 8
7027 LET t(i,j)=0
7029 NEXT j: NEXT i
7030 PRINT AT 17,5;"CARACTERE GR
APHIQUE ";b$;"  "
7040 FOR j=1 TO 8
7050 LET a=v(n*8+j): IF a=0 THEN
 GO TO 7120
7060 FOR i=1 TO 8
7070 LET t(j,i)=INT (a/2↑(8-i))
7080 PRINT AT j+3,i+11;" "
7090 IF t(j,i)=1 THEN LET a=a-2↑
(8-i): PRINT AT j+3,i+11;"■"
7100 NEXT i
```

```
7110 NEXT j
7120 NEXT j
7125 REM affinit
7130 LET x=8: LET y=8
7140 LET hx=8: LET hy=8
7150 PLOT 92,76
7160 DRAW INK 2,0,72: DRAW INK 2
,72,0
7170 DRAW INK 2,0,-72: DRAW INK
2,-72,0
7180 RETURN
7499 REM
7500 REM SAISIE
7501 REM
7510 LET a$=""
7520 LET a$=INKEY$: IF a$="" THE
N GO TO 7520
7530 IF CODE a$>96 THEN LET a$=C
HR$ (CODE a$-32)
7540 RETURN
7599 REM
7700 REM AFFDATA
7701 REM
7710 FOR j=1 TO 8
7720 PRINT AT j+3,21;v(n*8+j)
7730 NEXT j
7740 RETURN
7999 REM
8000 REM IMPOSSIBLE
8001 REM
8010 BEEP 0.1,10
8020 PRINT AT 21,0;"IMPOSSIBLE"
8030 PAUSE 50
8040 PRINT AT 21,0;"           "
8050 RETURN
8499 REM
8500 REM CHOIX
8501 REM
8510 PRINT AT 21,0;"Caractere ?"
8520 GO SUB saisie
8530 PRINT AT 21,0;"           "
8540 LET b$=a$
8550 IF CODE b$>68 OR CODE b$<65
 THEN GO TO 8510
8560 LET n=CODE b$-65
8570 RETURN
8999 REM
9000 REM PRESENTATION
9001 REM
9010 BORDER 0: PAPER 0: INK 4
9020 CLS
9030 DIM t(8,8): DIM v(32)
9040 FOR i=0 TO 7
9050 POKE USR "A"+i,0
9060 POKE USR "B"+i,0
9070 POKE USR "C"+i,0
9080 POKE USR "D"+i,0
9090 NEXT i
```

```
9100 LET c$=">COM: Init Def Voi
r Jux Fin "
9110 LET d$=">DEF: Chiff Dess
Fin"
9120 LET e$=">DES: Noir Eff Val
 Copy Fin "
9130 LET f$=">JUX: Dess Imp Fi
n "
9140 LET init=1000: LET definiti
on=2000
9150 LET voir=3000: LET jux=4000
9160 LET chiffre=5000: LET dessi
n=6000
9170 LET noir=6300: LET efface=6
400
9180 LET valide=6500: LET affich
age=7000
9190 LET saisie=7500: LET imposs
ible=8000
9200 LET choix=6600: LET affdata
=7700
9210 LET affinit=7130
9220 RETURN
```

1.6. REDÉFINITION DU GÉNÉRATEUR DE CARACTÈRES

Le ZX Spectrum possède un jeu de 96 caractères prédéfinis (codes ASCII compris entre 32 et 128), ce jeu étant placé dans la ROM entre les adresses 3DØØ en hexadécimal (15616 en décimal) et 3FFF (16383 en décimal). Un jeu complémentaire de vingt et un caractères, redéfinissables au gré de l'utilisateur, est disponible dans une partie de la RAM réservée à cet effet.

Nous avons vu précédemment comment procéder pour définir les caractères du jeu complémentaire. Bien que souvent suffisant, ce jeu complémentaire peut s'avérer être un peu limité dans certaines applications.

Nous allons donc voir maintenant comment redéfinir la totalité du générateur de caractères du ZX Spectrum, ce qui nous permettra d'avoir jusqu'à 96 caractères dessinés à notre volonté.

1.6.1. Principe

L'adresse de départ du jeu de caractères principal est contenue (à 256 près) dans les variables systèmes d'adresses 236Ø6 et 236Ø7. Cette adresse @ se calcule de la façon suivante :

@ =PEEK 236Ø6+256*PEEK 236Ø7+256

En fonctionnement normal, PEEK 236Ø6 = Ø et PEEK 236Ø7 = 6Ø, d'où @ = 15616 (adresse qui pointe sur la ROM).

A l'aide de l'instruction POKE, on peut changer les valeurs des variables système d'adresses 236Ø6 et 236Ø7 pour pointer sur une partie de la RAM où l'on pourra définir un nouveau jeu de caractères.

Avant de définir ce nouveau jeu de caractères, il convient tout d'abord de lui réserver un emplacement protégé de la RAM, en fixant avec l'instruction CLEAR l'adresse du dernier octet de la zone du système Basic. Par exemple CLEAR 31ØØØ permet de réserver suffisamment de place en RAM pour le nouveau jeu de caractères (sur ZX Spectrum 16 K ou 48 K).

Après avoir fait cela, nous sommes certains de disposer d'une zone de RAM protégée contre toute écriture de la part du système Basic et contre toute instruction dangereuse du type NEW.

Nous pouvons maintenant commencer par recopier l'ancien générateur de caractères dans le nouveau. Cela permet de disposer des caractères alphanumériques indispensables pour afficher des messages et de tous les autres caractères qui seront redéfinis pour les besoins des jeux.

Supposons donc que le nouveau jeu de caractères en RAM soit le duplicata exact du générateur de caractères de la ROM. Il est maintenant possible de redéfinir à volonté les 96 caractères du nouveau jeu en procédant comme pour les vingt et un caractères redéfinissables d'origine : codage sur une matrice 8 × 8, saisie des codes octet par octet.

L'adresse du premier des huit octets composants un caractère se calcule de la façon suivante :

@ = adresse de départ du nouveau jeu + 8 * (code ASCII du caractère − 32)

Exemple:

A code ASCII=65

adresse de départ du nouveau jeu: 31ØØØ

@ =31ØØØ+8*(65–32)=31264

A est donc codé par les octets 31264, 31265, ..., 31271.

Pour utiliser le nouveau jeu de caractères, il suffit de mettre les valeurs nécessaires dans les variables système d'adresse 236Ø6 et 236Ø7.

Exemple: Adresse de départ du nouveau jeu = 31ØØØ:

31ØØØ=12Ø*256+24+256

Il faut donc faire:

```
POKE 236Ø6,24
POKE 236Ø7,12Ø
```

Pour revenir au jeu normal de caractères, il faut faire:

```
POKE 236Ø6,Ø
POKE 236Ø7,6Ø
```

1.6.2. Exemple

Le programme ci-dessous résume tout ce qui a été dit précédemment au sujet de la redéfinition d'un jeu complet de caractères en RAM:

```
  10 CLEAR 31000
  20 FOR i=0 TO 767
  30 POKE 31000+i,PEEK (15616+i)
  40 NEXT i
  50 PRINT "nouveau generateur d
e caracteres": PRINT
  60 POKE 23606,24
  70 POKE 23607,120
  80 FOR i=32 TO 128
  90 PRINT CHR$ (i);
 100 NEXT i
 110 PRINT AT 10,5;"redefinition
de l'espace"
 120 FOR i=0 TO 7
 130 POKE 31000+i,255
 140 NEXT i
 150 PRINT AT 12,15;CHR$ (32)
 160 POKE 23606,0
 170 POKE 23607,60
```

La ligne 1Ø réserve la place mémoire nécessaire pour le nouveau jeu de caractères.

Les lignes 2Ø à 4Ø recopient l'ancien jeu de caractères dans le nouveau.

Les lignes 6Ø et 7Ø permettent de passer au nouveau jeu de caractères.

Les lignes 8Ø à 1ØØ affichent le nouveau jeu de caractères (identique pour l'instant à l'ancien).

Les lignes 12Ø à 14Ø changent le caractère de code ASCII 32 (espace) en un petit carré noir.

La ligne 15Ø affiche l'"espace" redéfini.

Les lignes 16Ø et 17Ø permettent de revenir au jeu normal de caractères.

1.7. GRAPHISMES ET SONS

Vous pourrez vous reporter utilement aux livres "La conduite du ZX Spectrum" et "Le grand livre du ZX Spectrum" (dans la collection Micro-ordinateur, Éditions Eyrolles), pour avoir des renseignements très complets sur les possibilités du ZX Spectrum en matière de couleurs, graphismes et sons. Le but de ce chapitre de "ZX Spectrum à la conquête des jeux" est simplement de vous montrer, sur quelques exemples, comment tirer parti des instructions graphiques et sonores du ZX Spectrum pour programmer des jeux.

1.7.1 Graphismes

Rappelons brièvement les caractéristiques des principales instructions graphiques du ZX Spectrum.

CIRCLE x,y,r : dessin d'un cercle de centre (x,y) et de rayon r.

PLOT x,y : imprime un point au pixel (x,y).

DRAW x,y : dessine une ligne à partir de la dernière position ayant été l'objet d'un PLOT en faisant un déplacement horizontal x et un déplacement vertical y.

DRAW x,y,r : même chose que l'instruction précédente si ce n'est que la ligne est courbée au lieu d'être droite (rayon de courbure r, sens de rotation direct ou indirect suivant le signe de r).

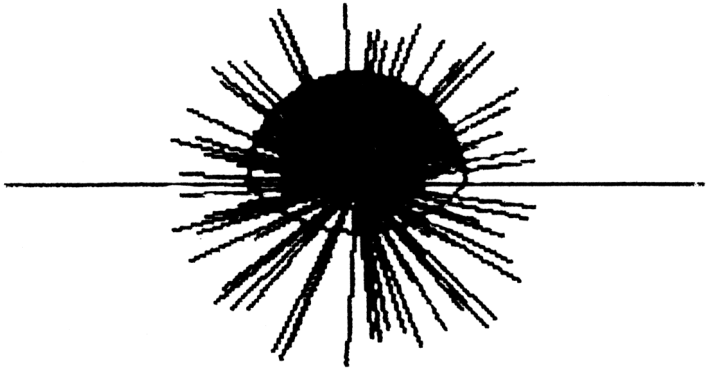
Les instructions INK, PAPER, FLASH, OVER et INVERSE peuvent être placées comme paramètres des instructions précédentes pour agir sur les attributs des points tracés : couleurs de l'encre et du papier, clignotement, surbrillance, ... Attention, les attributs concernent toujours une matrice 8 × 8 de points et non pas un pixel unique !

Les codes des couleurs sont :

Ø noir
1 bleu
2 rouge
3 bleu pâle
4 vert
5 bleu de prusse
6 jaune
7 blanc

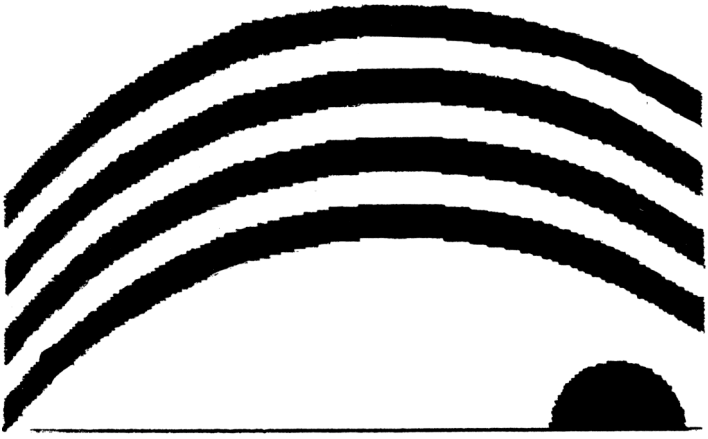
Après ces quelques rappels, voyons sur quelques exemples simples ce dont est capable le ZX Spectrum.

Exemple 1 : Coucher de soleil



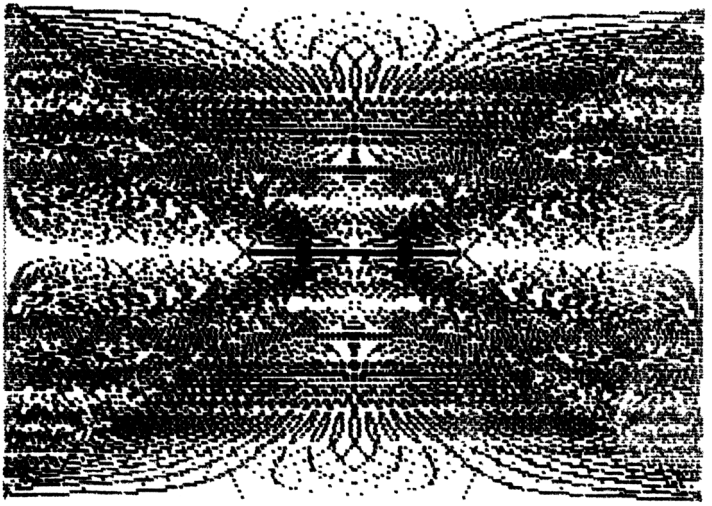
```
  10 BORDER 1: PAPER 1: INK 6: C
LS
  20 PLOT 88,80: DRAW 80,0,-PI
  30 PLOT 88,60: DRAW INK 3,80,0
 PI/2
  40 PLOT 0,80: DRAW INK 5,255,0
  50 FOR x=-40 TO 40
  60 PLOT 128,80: DRAW x,SQR (16
00-x*x)
  70 NEXT x
  80 FOR x=-70 TO 70 STEP 3
  90 LET y=INT (RND*SQR (4900-x*
x))
 100 PLOT 126,80: DRAW INK 3,x,-
y
 110 PLOT 126,80: DRAW x,y
 120 NEXT x
```

Exemple 2 : Arc-en-ciel



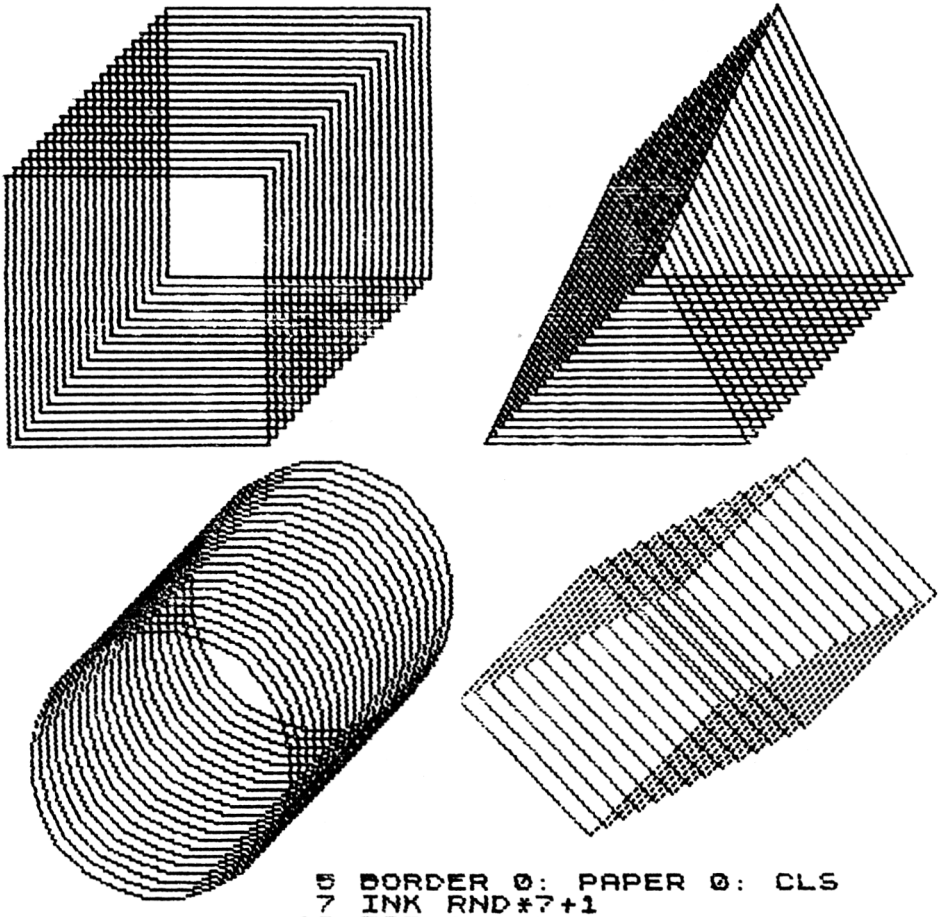
```
  10 BORDER 0: PAPER 0: INK 7: C
LS
  20 FOR i=0 TO 7
  30 READ a,b,c
  40 FOR j=0 TO 11*(i<7)
  50 PLOT PAPER b; INK c; INVERS
E a;0,j+i*12: DRAW PAPER b; INK
c; INVERSE a;255,35,-1.6
  60 IF ABS ((i/2)-INT (i/2))>0.
1 THEN LET j=j+7
  70 NEXT j
  80 NEXT i
  90 PAPER 0: INK 7: INVERSE 0
 100 FOR i=0 TO 48
 110 PLOT 200+i,0: DRAW INK 6;48
-i,0,-PI
 120 NEXT i
 130 PLOT INK 6;10,0: DRAW INK 6
;245,0
 140 STOP
 150 DATA 0,0,2,1,6,",0,6,4,1,5,
4,0,5,1,1,3,1,0,3,0,1,0,0
```

Exemple 3 : Chatoiement



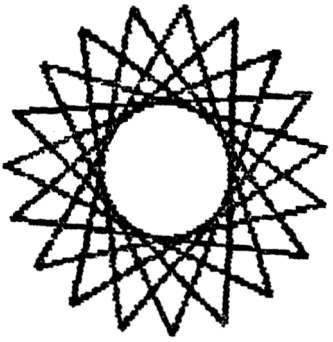
```
  10 BORDER 0: PAPER 0: INK 7
  20 BRIGHT 1: OVER 1
  30 CLS
  40 LET a=256
  50 FOR i=0 TO 1
  60 INK RND*7+1
  70 FOR j=0 TO 1
  80 FOR k=0 TO a-1
  90 PLOT 0,175*j: DRAW k,175-35
0*j
 100 PLOT a-1,175*j: DRAW -k,175
-350*j
 110 NEXT k
 120 NEXT j
 130 NEXT i
 140 GO TO 10
```

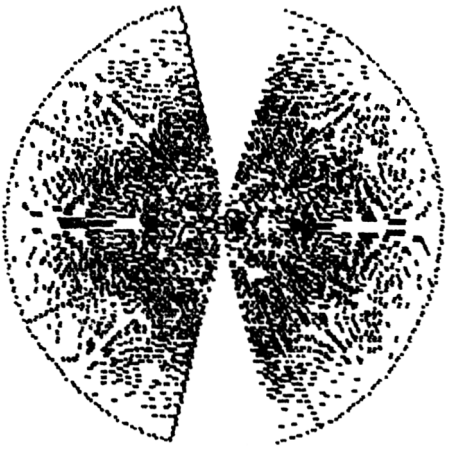
Exemple 4 : Géométrie



```
  5 BORDER 0: PAPER 0: CLS
  7 INK RND*7+1
 10 FOR i=0 TO 60 STEP 3
 20 PLOT 64+i,8+i
 30 DRAW 96,0: DRAW 0,96
 40 DRAW -96,0: DRAW 0,-96
 50 NEXT i
 60 CLS
 70 FOR i=0 TO 60 STEP 3
 80 CIRCLE 112+i,56+i,48
 90 NEXT i
100 CLS
110 FOR i=0 TO 60 STEP 3
120 PLOT 64+i,8+i
130 DRAW 48,96: DRAW 48,-96
140 DRAW -96,0
150 NEXT i
155 CLS
160 FOR i=0 TO 40 STEP 3
170 PLOT 64+i*2,56+i
180 DRAW 48,48: DRAW 48,-48
190 DRAW -48,-48: DRAW -48,48
200 NEXT i
210 CLS
220 GO TO 7
```

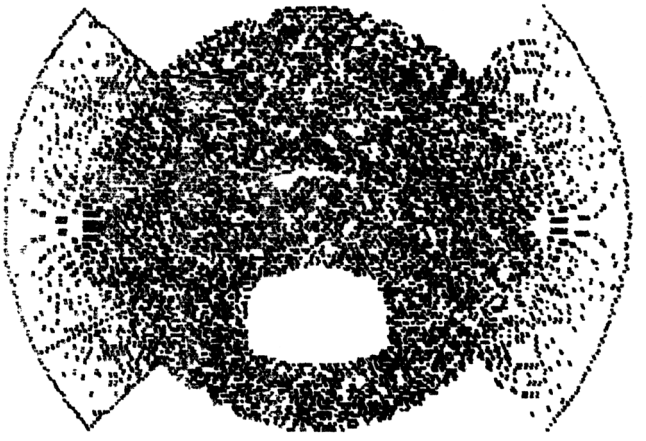
Exemple 5 : Particularités de l'instruction DRAW





```
  10 BORDER 0: PAPER 0: INK 4: C
LS
  20 PLOT 120,10
  30 DRAW INK RND*7+1; OVER 1;0,
155,8705.835
  40 PLOT 190,60
  50 DRAW INK RND*7+1;0,55,1000
  50 GO TO 20
```

Exemple 6: Présentation du jeu EXPLORATION SPATIALE



```
  10 BORDER 0: PAPER 0: INK 4: C
LS
  20 PLOT 50,0
  30 DRAW OVER 1, INK 5,0,155,87
07
  40 FOR i=5 TO 1 STEP -1
  50 PLOT 128,0
  60 DRAW OVER 1, INK 2,i*1.5,17
5-i*20,2771+PI
  70 NEXT i
  80 CIRCLE FLASH 1,88,88,10
  90 CIRCLE FLASH 1,175,88,10
```

Exemple 7: Comment faire un joli dessin en peu de lignes

```
10 PLOT 128,0
20 DRAW OVER 1,0,155,2771*PI
30 PLOT 179,50
40 DRAW 0,55,1000
```

1.7.2 Sons

Les possibilités sonores du ZX Spectrum sont bien faibles en comparaison de ce que proposent dans ce domaine d'autres micro-ordinateurs de la même catégorie.

Une seule instruction permet d'accéder à partir du Basic au haut-parleur interne: BEEP x,y où x règle la durée du son émis et y règle sa hauteur.

L'instruction OUT 254,n permet aussi d'obtenir des sons mais elle agit aussi sur la couleur du bord de l'écran, ce qui n'est pas toujours souhaitable.

On peut malgré tout s'efforcer de créer des effets sonores pour les jeux: tir d'un missile, sirène américaine, ... Vous trouverez ci-dessous quelques exemples qui vous donneront peut être des idées:

```
10 FOR i=1 TO 30
20 BEEP 0.01,20
30 BEEP 0.01,30
40 BEEP 0.01,10
50 NEXT i
```

```
10 FOR j=1 TO 5
20 FOR i=10 TO 30
30 BEEP 0.003,i
40 NEXT i
50 NEXT j
```

```
10 FOR J=1 TO 3
20 FOR i=10 TO 30
30 BEEP 0.003,i
40 BEEP 0.003,40-i
50 NEXT i
60 NEXT j
```

1.8. ROUTINE DE RENUMÉROTATION

Le petit programme ci-dessous vous permettra de renuméroter très aisément vos programmes Basic. Son principe repose sur le mode de stockage des lignes de Basic en mémoire.

Attention, ce programme de renumérotation n'agit pas sur les paramètres de GOSUB et GOTO. Il vous faudra donc changer vous-même tous ces paramètres après avoir effectué une renumérotation des lignes de votre programme.

```
9976 CLS : PRINT "RENUM ";
9977 INPUT "Nbre de lignes a ren
umeroter ",n
9978 IF n=0 THEN GO TO 9977
9979 PRINT n;",";
9980 INPUT "Numero ligne de depa
rt ",lignedep
9981 IF lignedep=0 THEN GO TO 99
80
9982 PRINT lignedep;",";
9983 INPUT "Pas ",pas
9984 IF pas=0 THEN GO TO 9983
9985 PRINT pas
9986 LET d=PEEK 23635+256*PEEK 2
3636
9987 LET ligne=lignedep
9988 GO SUB 9996
9989 FOR i=1 TO n-1
9990 LET d=d+4+PEEK (d+2)+256*PE
EK (d+3)
9991 LET ligne=ligne+pas
9992 GO SUB 9996
9993 NEXT i
9994 BEEP 0.01,30: PRINT "OK"
9995 STOP
9996 LET x=INT (ligne/256)
9997 LET y=ligne-256*x
9998 POKE d,x: POKE d+1,y
9999 RETURN
```

2

Trucs et astuces pour la programmation de jeux en langage machine sur ZX Spectrum

2.1. GÉNÉRALITÉS

L'intérêt de la programmation en langage machine n'est un secret pour personne. Tous les programmeurs ont été limités au moins une fois dans leur vie par la lenteur d'exécution et par le manque de souplesse du Basic, qui restreint les possibilités d'un micro-ordinateur aux seules caractéristiques des quelques instructions dont il dispose.

Contrairement au Basic, le langage machine est extrêmement rapide à l'exécution et, grâce à lui, on peut espérer tirer parti au mieux de son ZX Spectrum.

Ce chapitre de "ZX Spectrum à la conquête des jeux" ne peut pas et ne veut pas être un cours complet sur la programmation en langage machine Z 8Ø du ZX Spectrum. Pour des renseignements très complets sur ce sujet, reportez-vous au livre "Trucs et astuces en langage machine sur ZX Spectrum" (Pascal Pellier, Éditions Eyrolles).

Vous trouverez dans les paragraphes suivants un bref survol de quelques points particuliers : accès au langage machine, carte mémoire, routines principales de la ROM, ... L'objectif de ce chapitre est de vous permettre d'insérer très simplement des petits sous-programmes en langage machine dans les programmes de jeux que vous pouvez être amené à écrire.

2.2. ACCÈS AU LANGAGE MACHINE

L'accès au langage machine sur ZX Spectrum peut se faire de trois façons :

- Vous disposez d'un programme moniteur qui vous permet d'entrer octet par octet les codes constituant votre programme, de faire de la mise au point, ...

- Vous possédez l'éditeur/assembleur pour ZX Spectrum (Logilivre aux Éditions Eyrolles) et dans ce cas vous n'avez plus aucun problème pour développer très facilement tous vos programmes en Z 8Ø sur ZX Spectrum. Il faut d'ailleurs remarquer qu'un outil tel que cet éditeur/assembleur est indispensable pour travailler efficacement en langage machine.

- Vous ne possédez qu'un ZX Spectrum standard et vous en êtes réduit à utiliser le langage machine par l'intermédiaire du Basic (POKE, USR, ...). Nous supposerons dans toute la suite du chapitre que vous êtes dans ce cas.

La première chose à faire, avant de tenter de stocker un programme en langage machine dans la RAM, est de s'assurer qu'un emplacement protégé contre toute écriture intempestive de la part du système Basic est bien disponible. Cela se fait très simplement à l'aide de l'instruction CLEAR qui agit sur la variable système RAMTOP et qui fixe ainsi l'adresse du dernier octet accessible par le système Basic.

Il faut ensuite entrer le programme en mémoire. La seule façon de procéder est d'utiliser l'instruction POKE. Vous trouverez dans le paragraphe suivant un petit programme de chargement de code hexadécimal.

Le code étant chargé, il ne reste plus qu'à lancer son exécution avec l'instruction Basic USR n où n est l'adresse de départ du programme.

REMARQUE : Si vous désirez revenir au Basic après l'exécution d'un programme en langage machine, il est indispensable que la dernière instruction de ce programme soit un RET.

L'instruction USR s'utilise généralement de la façon suivante :

PRINT USR n : lance l'exécution du sous-programme et affiche au retour le contenu en décimal de la paire de registre BC.

RANDOMIZE USR n : lance l'exécution du sous-programme et agit sur le générateur de nombres aléatoires.

LET A = USR n : lance l'exécution du sous-programme et récupère dans la variable A le contenu en décimal de la paire de registres BC.

2.3. CHARGEMENT DE CODE HEXADÉCIMAL

Le programme donné ci-dessous n'a bien sûr rien de comparable avec le merveilleux outil de programmation en langage machine qu'est l'éditeur/assembleur pour ZX Spectrum proposé par les Éditions Eyrolles. Il a cependant pour lui d'être simple et court.

Le but de ce programme est de vous permettre d'entrer en mémoire les codes hexadécimaux du programme que vous aurez écrit puis assemblé à la main... Ces codes sont placés dans des DATA et le programme de chargement va les "POKer" un à un aux emplacements voulus. Vous pourrez ensuite lancer l'exécution de votre programme en faisant PRINT USR 32000 ou RANDOMIZE USR 32000 ou LET variable = USR 32000 (l'adresse 32000 a été choisie parce qu'elle convient bien aux ZX Spectrum 16 K et 48 K mais vous pouvez en choisir une autre).

N'oubliez pas que, pour revenir au Basic en fin d'exécution, vous devez avoir mis un RET comme dernière instruction de votre programme.

```
  10 CLEAR 32000
  20 LET d=32000
  30 DEF FN a(a$,i)=CODE a$(i)-4
8-7*(CODE a$(i)>57)
  40 DEF FN b(a$)=16*FN a(a$,1)+
FN a(a$,2)
  50 READ a$
  60 IF a$<>"**" THEN POKE d,FN
b(a$): LET d=d+1: GO TO 50
  70 STOP
```

La ligne 1Ø protège toute la partie de RAM située après l'adresse 32ØØØ.

Les lignes 3Ø et 4Ø définissent deux fonctions qui sont utilisées pour convertir la chaîne de caractères que constitue un code hexadécimal en un entier compris entre Ø et 255.

La ligne 5Ø lit les DATA, code par code.

La ligne 6Ø "poke" le code lu à l'adresse d jusqu'à ce que ce code lu soit "**" signalant ainsi la fin du programme.

Exemple d'application:

```
  10 CLEAR 32000
  20 LET d=32000
  30 DEF FN a(a$,i)=CODE a$(i)-4
8-7*(CODE a$(i)>57)
  40 DEF FN b(a$)=16*FN a(a$,1)+
FN a(a$,2)
  50 READ a$
  60 IF a$<>"**" THEN POKE d,FN
b(a$): LET d=d+1: GO TO 50
  70 STOP
  80 DATA "21","00","04","11","0
1","00","E5","D5","CD","85","03"
,"D1","E1","01","08","00","AF","
ED","42","30","F1","C9","**"
```

Après avoir exécuté ce programme, faites en mode direct:

FOR a=1 TO 1Ø: RANDOMIZE USR 32ØØØ: NEXT a

et écoutez bien!

2.4. AFFICHAGE DE CARACTÈRES SUR L'ÉCRAN

Pour tout affichage de caractères sur l'écran, le ZX Spectrum utilise l'instruction RST ØØ1ØH qui fait appel à la routine PRINT-OUTPUT située en ROM à l'adresse Ø9F4H. Cette routine sera utilisée dans le sens OUTPUT si le canal 'S' a été ouvert au préalable par un appel à la routine CHAN-OPEN située en ROM à l'adresse 16Ø1H.

L'instruction RST ØØ1ØH est particulièrement puissante puisqu'elle permet non seulement d'afficher n'importe quel caractère (comme PRINT) mais aussi de changer la position courante d'affichage (AT, TAB), de choisir des attributs (PAPER, INK, FLASH, ...).

Rappelons les codes hexadécimaux de certaines instructions :

Instruction	*Code hexadécimal*
PRINT,	Ø6
ENTER	ØD
INK	1Ø
PAPER	11
FLASH	12
BRIGHT	13
INVERSE	14
OVER	15
AT	16
TAB	17

Le programme ci-dessous donne quelques exemples d'utilisation de ces codes et de RST ØØ1ØH :

```
; ouverture du canal S
    LD   A,Ø2
    CALL CHAN-OPEN

; simulation de PRINT AT 1Ø,5; 'A'
    LD   A,16H      ; charge le code hexadécimal de AT
    RST  ØØ1ØH
    LD   A,1Ø       ; charge position verticale (Y=1Ø)
    RST  ØØ1ØH
    LD   A,Ø5       ; charge position horizontale (X=5)
    RST  ØØ1ØH
    LD   1,41H      ; charge le code de 'A'
    RST  ØØ1ØH

; simulation de PRINT
    LD   A,ØDH      ; charge le code de ENTER
    RST  ØØ1ØH

; simulation de PRINT INK 2; 'B'
    LD   A,1Ø       ; charge le code de INK
    RST  ØØ1ØH
    LD   A,Ø2       ; charge le code de la couleur rouge
    RST  ØØ1ØH
    LD   A,42H      ; charge le code de 'B'
    RST  ØØ1ØH

; retour au Basic
    RET
```

Une fois assemblé, ce programme donne une liste de codes qui peuvent être mis dans des DATA lus par le chargeur donné en 2.2.

D'où la ligne 8Ø:

```
  8Ø>DATA "3E","Ø2","CD","Ø1","1
5","3E","16","D7","3E","1Ø","D7"
,"3E","Ø5","D7","3E","41","D7","
3E","ØD","D7","3E","1Ø","D7","3E
","Ø2","D7","3E","42","D7","C9",
"**"
```

L'exécution de ce programme que vous obtiendrez en faisant RANDOMIZE USR 32ØØØ (après avoir utilisé le chargeur de code) donnera un résultat semblable à ce qu'aurait produit le programme Basic suivant:

```
1Ø PRINT AT 1Ø,5; "A"
2Ø PRINT
3Ø PRINT INK 2; "B"
```

2.5. EFFACEMENT DE L'ÉCRAN

Effacer la totalité de l'écran et redonner à tous les attributs leurs valeurs "permanentes" est réalisé par la routine CLS située en ROM à l'adresse ØD6BH.

De même que pour la routine PRINT-OUTPUT, il faut avoir ouvert le canal S avant d'utiliser la routine CLS. Il faudra ensuite le réouvrir si des caractères doivent être affichés.

Exemple:

```
LD A,Ø2                 ; ouverture du canal S
CALL CHAN-OPEN
CALL CLS                ; effacement de l'écran
LD A,Ø2                 ; réouverture du canal S
CALL CHAN-OPEN
RET
```

L'intérêt du langage machine réside surtout dans la possibilité d'employer la routine CL-LINE, située dans la ROM à l'adresse ØE44H, qui permet d'effacer seulement un nombre spécifié de lignes. Là aussi les

attributs concernant les lignes effacées reprendront leurs valeurs permanentes.

Avant d'appeler la routine CL-LINE, il faut placer dans le registre B la valeur hexadécimale correspondant au nombre de lignes à effacer (valeur comprise entre ØlH et 18H) à partir du bas de l'écran.

Exemple :

```
LD B,1ØH                ; 16 lignes vont être effacées
CALL CL-LINE
RET
```

Rappel: CHAN-OPEN = 16Ø1H.

2.6. SCROLLING DE L'ÉCRAN

La routine CL-SCROLL, située en ROM à l'adresse ØEØØH, permet de "scroller" vers le haut un nombre spécifié de lignes de l'écran, ce qui est particulièrement intéressant pour certains jeux.

Le registre B doit contenir le nombre de lignes à "scroller" sachant que ce nombre doit être supérieur ou égal à 2.

Exemple :

```
LD A,16H
CALL CL-SCROLL
RET
```

REMARQUE : Ni la routine CL-LINE, ni la routine CL-SCROLL ne nécessitent l'ouverture du canal S.

2.7. UTILISATION DU GÉNÉRATEUR DE SONS

Si vous avez utilisé le programme d'exemple donné dans le paragraphe 2.2, vous avez déjà vu ce qu'il est possible d'obtenir en matière d'effets sonores avec le ZX Spectrum programmé en langage machine.

La routine BEEPER, située dans la ROM à l'adresse Ø3B5H, permet de produire des sons dont la fréquence est conditionnée par la

valeur contenue dans la paire de registres HL, et la hauteur par la valeur contenue dans DE.

Pour obtenir un son de fréquence f, il faut placer dans HL, la valeur de l'expression 437500/f – 30,125 et dans DE la valeur de f * t où t est la durée désirée.

Exemple:

```
LD DE,0105H
LD HL,0666H
CALL BEEPER
RET
```

2.8. REDÉFINITION DU GÉNÉRATEUR DE CARACTÈRES

Nous avons vu dans le chapitre consacré à la programmation des jeux en Basic, comment redéfinir le générateur de caractères du ZX Spectrum. Vous avez certainement constaté que le programme d'exemple en Basic était très lent pour recopier l'ancien jeu de caractères dans le nouveau.

Or cela se fait très simplement et très rapidement en utilisant l'instruction LDIR du Z 80, d'où le programme suivant:

```
  10 CLEAR 31000
  20 FOR i=0 TO 11
  30 READ a
  40 POKE 32000+i,a
  50 NEXT i
  60 DATA 33,0,61,17,24,121,1,0,
0,237,176,201
  70 RANDOMIZE USR 32000
  80 POKE 23606,24
  90 POKE 23607,120
 100 PRINT "nouveau generateur d
e caracteres": PRINT
 110 FOR i=32 TO 128
 120 PRINT CHR$ (i);
 130 NEXT i
 140 PRINT AT 10,5;"redefinition
de l'espace"
 150 FOR i=0 TO 7
 160 POKE 31000+i,255
 170 NEXT i
 180 PRINT AT 12,15;CHR$ (32)
 190 POKE 23606,0
 200 POKE 23607,60
```

Ce chapitre n'était qu'une brève initiation à la programmation en langage machine sur ZX Spectrum. Vous connaissez maintenant quelques unes des principales routines de la ROM avec leurs points d'entrée et les paramètres requis. Vous pouvez donc d'ores et déjà commencer à écrire vos propres jeux en Z 8Ø.

Pour plus de détails sur la programmation du Z 8Ø, reportez-vous au livre "L'assembleur facile du Z 8Ø" (Éditions Eyrolles). Vous découvrirez tous les trésors cachés du ZX Spectrum dans "Trucs et astuces en langage machine" (Éditions Eyrolles).

3

Jeux

3.1. ALPHABET

ALPHABET est un jeu de réflexion qui vous donnera bien des soucis car il est très difficile.

Imaginez en effet un tableau contenant les vingt-quatre premières lettres de l'alphabet dans le désordre le plus complet. Le seul degré de liberté, qui vous est laisse pour remettre ces lettres dans le bon ordre, vous est donné par un curseur clignotant. A l'aide des touches 5, 6, 7 et 8 (la manette de jeu 1 est donc aussi utilisable), vous pouvez déplacer ce curseur clignotant pour échanger sa position avec celle d'une des lettres qui lui sont adjacentes.

Après un écran de présentation du jeu, le programme vous montre le résultat à obtenir tandis qu'il prépare le tableau de jeu initial.

```
voila le resultat que vous devez
obtenir
```

Ce tableau étant enfin prêt, à vous de jouer... et bonne chance.

```
COUP:  0

                A F K P U
                B G L M V
                C H I R Q
                D O N S X
                E J W T


A vous...   ('0' pour arreter)
```

Structure du programme

- *Programme principal* : lignes 1 à 1ØØ

Ligne 1Ø : appel du sous-programme de présentation.
Ligne 2Ø : appel du sous-programme de mélange et d'initialisation.
Ligne 3Ø : appel du sous-programme de jeu.
Ligne 4Ø : cas où le joueur a gagné.
Lignes 45 à 49 : routine de sonorisation.
Lignes 5Ø et 6Ø : attente de la pression d'une touche avant de continuer.
Lignes 7Ø à 9Ø : rechargement du tableau initial non mélangé.
Ligne 1ØØ : retour au début du jeu.

- *Sous-programme de permutation de lettres* : lignes 1ØØØ à 1Ø3Ø

Lignes 1ØØØ à 1Ø3Ø : permutation de deux lettres du tableau a$ en utilisant une variable intermédiaire b$.

- *Sous-programme d'affichage du tableau de lettres* : lignes 2ØØØ à 2Ø7Ø

Lignes 2ØØØ à 2Ø6Ø : affichage ligne après ligne du tableau de lettres a$ (ligne 2Ø2Ø) en prenant soin de faire clignoter la case vide (ligne 2Ø3Ø).

Ligne 2Ø7Ø : retour au sous-programme appelant (initialisation ou jeu suivant le cas).

- *Sous-programme de jeu* : lignes 3ØØØ à 313Ø

Ligne 3ØØØ : lecture du clavier jusqu'à ce qu'une touche soit pressée.

Ligne 3ØØ5 : possibilité d'abandon en cours de partie en appuyant sur la touche Ø (ou le bouton rouge de la manette 1). Le drapeau FIN est mis à 2 pour indiquer qu'il s'agit là d'un abandon et non pas d'une victoire.

Ligne 3Ø1Ø : le compteur de coups est incrémenté puis affiché en haut de l'écran.

Lignes 3Ø2Ø et 3Ø3Ø : calcul de la position de la lettre qui va se mettre à la place de la case vide.

Ligne 3Ø4Ø : appel du sous-programme de permutation.

Lignes 3Ø6Ø et 3Ø65 : affichage de la lettre échangée (ligne 3Ø6Ø) et de la case vide clignotante (ligne 3Ø65).

Ligne 3Ø8Ø : le drapeau de fin est mis à 1 (victoire).

Lignes 3Ø9Ø à 311Ø : le tableau de référence e$ et le tableau de jeu a$ sont comparés lettre à lettre. Dès qu'une différence apparaît, le drapeau FIN est mis à Ø.

Ligne 312Ø : test du drapeau FIN. Le jeu continu si ce drapeau est à Ø alors que la victoire est signifiée par FIN à 1.

Ligne 313Ø : retour au programme principal en cas de victoire.

- *Sous-programme de mélange et d'initialisation* : lignes 4ØØØ à 417Ø

Ligne 4ØØØ : effacement de l'écran. Mise à zéro du nombre de coups joués.

Ligne 4Ø1Ø : appel du sous-programme d'affichage pour présenter le tableau de lettres initial.

Ligne 4Ø2Ø : initialisation de la position de la case vide.

Lignes 4Ø5Ø à 412Ø : mélange du tableau a$.

Ligne 413Ø : effacement de l'écran.

Ligne 414Ø : affichage du tableau en désordre.

Ligne 417Ø : retour au programme principal.

- *Sous-programme de présentation* : lignes 5ØØØ à 54ØØ

Ligne 5ØØØ : l'écran et son bord seront de couleur noire alors que l'encre des caractères sera verte. Effacement de l'écran.

Ligne 5Ø1Ø : déclaration du tableau t$ utilisé pour la présentation, du tableau de jeu a$ qui contiendra les lettres en désordre et du tableau de référence e$.

Lignes 5Ø3Ø à 51ØØ : initialisation des tableaux a$ et e$.

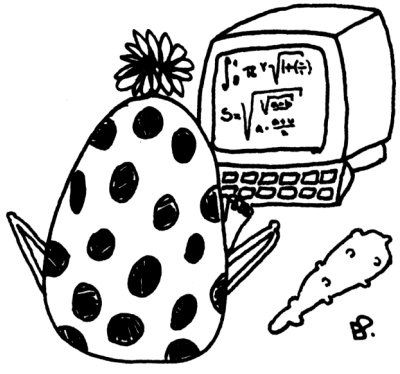
Lignes 516Ø à 53ØØ : initialisation du tableau t$ avec les caractères composant ligne après ligne les lettres du nom du jeu : ALPHABET.

Lignes 531Ø à 536Ø : affichage ligne après ligne du nom du jeu en grosses lettres colorées.

Lignes 537Ø et 538Ø : attente de la pression d'une touche pour continuer.

Ligne 539Ø : effacement de l'écran.

Ligne 54ØØ : retour au programme principal.



```
   1 REM ************
   2 REM * ALPHABET *
   3 REM ************
   4 REM
  10 GO SUB 5000
  20 GO SUB 4000
  30 GO SUB 3000
  40 IF fin=1 THEN PRINT AT 15,3
;"vous avez gagne en ";coup;" co
ups"
  45 FOR i=1 TO 10
  46 FOR j=1 TO 10
  47 BEEP 0.005,j
  48 NEXT j
  49 NEXT i
  50 PRINT AT 21,0;"pressez une
touche pour rejouer"
  60 PAUSE 4E4: PAUSE 4E4: CLS
  70 FOR i=1 TO 5
  80 LET a$(i)=e$(i)
  90 NEXT i
 100 GO TO 20
1000 LET b$=a$(y,x)
1010 LET a$(y,x)=a$(hy,hx)
1020 LET a$(hy,hx)=b$
1030 RETURN
2000 FOR j=1 TO 5
2010 FOR k=1 TO 5
2020 PRINT AT 2*j+1,2*k+9; INK I
NT (RND*7+1);a$(j,k)
2030 IF a$(j,k)=" " THEN PRINT A
T 2*j+1,2*k+9; FLASH 1;" ": LET
x=k: LET y=j
2040 NEXT k
2050 REM
2060 NEXT j
2070 RETURN
3000 LET c$=INKEY$: IF c$="" THE
N GO TO 3000
3005 IF c$="0" THEN LET fin=2: R
ETURN
3010 LET coup=coup+1: PRINT AT 0
,0;"COUP: ";coup
3020 LET y=y+(c$="6" AND y<5)-(c
$="7" AND y>1)
3030 LET x=x+(c$="8" AND x<5)-(c
$="5" AND x>1)
3040 GO SUB 1000
3060 PRINT AT 2*hy+1,2*hx+9; INK
 INT (RND*7+1);a$(hy,hx)
3065 PRINT AT 2*y+1,2*x+9; FLASH
 1;a$(y,x): BEEP 0.1,20
3070 LET hx=x: LET hy=y
3080 LET fin=1
3090 FOR i=1 TO 5
3100 IF a$(i)<>e$(i) THEN LET fi
n=0
3110 NEXT i
3120 IF fin=0 THEN GO TO 3000
3130 RETURN
```

```
4000 CLS : LET coup=0
4010 GO SUB 2000
4020 LET hx=5: LET hy=5
4030 PRINT AT 11,19; FLASH 1;" "
4040 PRINT AT 19,0;"voila le res
ultat que vous devezobtenir"
4050 FOR i=1 TO 50
4060 LET x=hx+2*INT (RND*2)-1
4070 IF (x-5)*(x-1)>0 THEN GO TO
 4060
4080 LET y=hy+2*INT (RND*2)-1
4090 IF (y-5)*(y-1)>0 THEN GO TO
 4080
4100 GO SUB 1000
4110 LET hx=x: LET hy=y
4120 NEXT i
4130 CLS
4140 GO SUB 2000
4150 PRINT AT 21,0;"A vous...  (
'0' pour arreter)"
4160 PRINT AT 0,0;"COUP: ";coup
4170 RETURN
5000 BORDER 0: PAPER 0: INK 4: C
LS
5010 DIM t$(15,32): DIM a$(5,5):
 DIM e$(5,5)
5030 LET a$(1)="AFKPU"
5040 LET a$(2)="BGLQV"
5050 LET a$(3)="CHMRW"
5060 LET a$(4)="DINSX"
5070 LET a$(5)="EJOT "
5080 FOR i=1 TO 5
5090 LET e$(i)=a$(i)
5100 NEXT i
5160 LET t$(1)="    L         H  H
   BBB     TTTT"
5170 LET t$(2)=" AA L   PPP H  H
 AA B  BEEEE TT "
5180 LET t$(3)="A  AL   P  PH  H
A  AB  BE    TT "
5190 LET t$(4)="A  AL   P  PH  H
A  AB  BE    TT "
5200 LET t$(5)="A  AL   P  PH  H
A  AB  BE    TT "
5210 LET t$(6)="A  AL   P  PH  H
A  AB  BE    TT "
5220 LET t$(7)="A  AL   PPP H  H
A  AB  BE    TT "
5230 LET t$(8)="A  AL   P   HHHH
A  ABBB E    TT "
5240 LET t$(9)="AAAAL   P   H  H
AAAAB  BEEE  TT "
5250 LET t$(10)="A  AL   P   H
HA  AB  BE    TT "
5260 LET t$(11)="A  AL   P   H
HA  AB  BE    TT "
5270 LET t$(12)="A  AL   P   H
HA  AB  BE    TT "
5280 LET t$(13)="A  AL   P   H
HA  AB  BE    TT "
```

```
5290 LET t$(14)="A   ALLLLP    H
HA   ABBB E      TT  "
5300 LET t$(15)="A   A      P
 A   A      EEEE    "
5310 FOR i=1 TO 15
5320 FOR j=0 TO 31
5325 LET c=INT (j/4)+1: IF c>=8
THEN LET c=1
5340 PRINT AT i+2,j; INK c;t$(i,
j+1)
5350 NEXT j
5360 NEXT i
5370 PRINT AT 21,0;"pressez ENTE
R pour commencer"
5380 PAUSE 4E4
5390 CLS
5400 RETURN
```

3.2. CHASE (VERSION CLAVIER)

CHASE est à la fois un jeu d'adresse et de réflexion. Deux joueurs s'affrontent dans un espace limité à l'écran. Les concurrents disposent d'un véhicule rapide qui laisse derrière lui une traînée. Les joueurs doivent se déplacer constamment en évitant tous les obstacles constitués par les deux traînées et les limites du terrain de jeu.

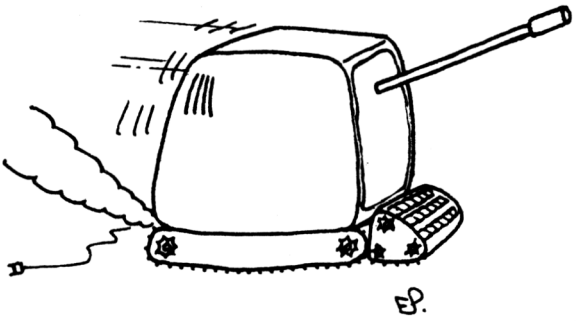
Il est impossible de s'arrêter et interdit de revenir sur ses pas. Dans ces conditions, le terrain se recouvrant très rapidement d'obstacles dangereux, la collision est inévitable. Le joueur le plus habile sera celui qui aura su provoquer la faute de son adversaire avant d'avoir été lui-même contraint à percuter un obstacle.

Lors de la présentation du jeu, le ZX Spectrum vous demande le nombre de manches qui vont être disputées. Dès que vous avez répondu à cette question, le compte à rebours commence et c'est parti...

Le joueur, qui contrôle le véhicule blanc, dispose des touches A, S, D et F qui ont exactement les mêmes fonctions pour ce jeu que les touches 5 (←), 6 (↓), 7 (↑) et 8 (→). Le joueur, qui possède le véhicule rouge, dispose des touches U, I, O et P.

Après une collision, le véhicule ayant percuté clignote. Pressez une touche quelconque pour voir le score. Repressez une touche pour lancer une nouvelle manche.

Pour rejouer en fin de partie, répondez par une simple pression de ENTER. Appuyez sur n puis sur ENTER pour arrêter le jeu.



Structure du programme

- *Programme principal* : lignes 1 à 76Ø

Ligne 5 : appel du sous-programme de présentation.

Ligne 1Ø : début de la boucle de jeu qui sera parcourue n fois, n étant le nombre de manches demandé par les joueurs lors de la présentation.

Ligne 2Ø : appel du sous-programme d'initialisation.

Ligne 3Ø : choix du joueur qui part en premier.

Ligne 11Ø : lecture de la demi-rangée ASDF affectée au joueur qui contrôle le véhicule blanc. Si ce joueur n'a pressé aucune touche, son véhicule continue d'avancer dans la direction qu'il avait déjà.

Lignes 12Ø et 13Ø : calcul de la nouvelle position du véhicule blanc.

Ligne 14Ø : teste s'il y a collision avec un obstacle quel qu'il soit.

Ligne 15Ø : sauvegarde de la direction prise par le véhicule blanc.

Ligne 16Ø : affichage du véhicule blanc.

Ligne 21Ø : lecture de la demi-rangée UIOP affectée au joueur qui contrôle le véhicule rouge. Si aucune touche n'est pressée, le véhicule rouge garde son ancienne direction.

Lignes 22Ø et 23Ø : calcul de la nouvelle position du véhicule rouge.

Ligne 24Ø : teste s'il y a collision avec un obstacle.

Ligne 25Ø : sauvegarde de la direction prise par le véhicule rouge.

Ligne 26Ø : affichage du véhicule rouge.

Ligne 27Ø : retour en ligne 1ØØ pour un nouveau déplacement des deux véhicules.

Lignes 5ØØ à 52Ø : cas où le joueur blanc a gagné. La voiture rouge clignote à l'endroit où elle vient de heurter un obstacle. Appel du sous-programme de sonorisation.

Lignes 6ØØ et 61Ø : cas où le joueur rouge gagne. La voiture blanche clignote. Appel du sous-programme de sonorisation.

Ligne 7ØØ : attente de la pression d'une touche.

Lignes 71Ø et 72Ø : effacement de l'écran et affichage des scores.

Ligne 73Ø : passage à la manche suivante dès qu'un joueur appuie sur une touche.

Ligne 74Ø : proposition de rejouer à la fin de la partie.

Ligne 75Ø : toute réponse commençant par la lettre n arrête le jeu.

Ligne 76Ø : arrêt du jeu.

- *Sous-programme de sonorisation* : lignes 8ØØ à 85Ø

- *Sous-programme d'initialisation* : lignes 1ØØØ à 114Ø

Ligne 1ØØØ : effacement de l'écran.

Lignes 1Ø1Ø et 1Ø2Ø : initialisation des positions de départ des véhicules.

Ligne 1Ø3Ø : choix des directions initiales des deux voitures.

Ligne 1Ø4Ø : la variable a$ est utilisée pour l'affichage du terrain de jeu.

Lignes 1Ø5Ø à 1Ø8Ø : affichage du terrain de jeu.

Lignes 1Ø9Ø à 113Ø : compte à rebours.

Ligne 116Ø : retour à la boucle de jeu.

- *Sous-programme de présentation* : lignes 2ØØØ à 215Ø

Ligne 2ØØØ : l'écran et son bord seront de couleur noire, l'encre des caractères sera verte. Effacement de l'écran.

Ligne 2Ø1Ø : déclaration du tableau qui contient les scores des deux joueurs.

Lignes 2Ø2Ø à 2Ø6Ø : définition du caractère graphique "A" qui représente les voitures.

Lignes 2Ø7Ø : affichage en grosses lettres multicolores du nom du jeu.

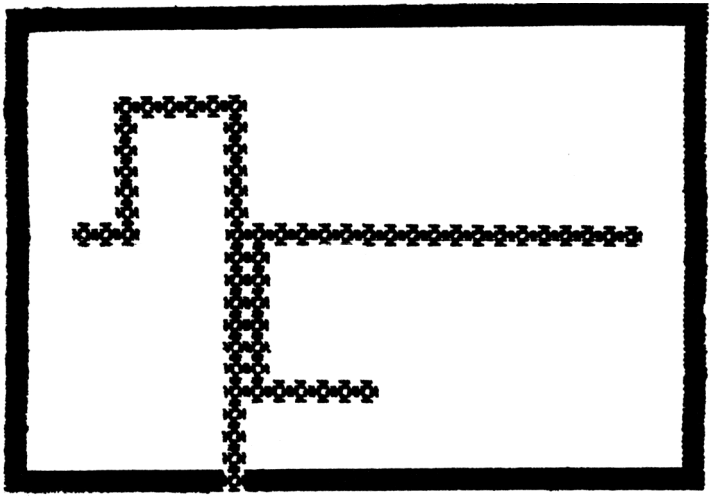
Ligne 212Ø: affichage de deux véhicules clignotants.

Ligne 213Ø: demande du nombre de manches qui vont être disputées.

Ligne 214Ø: protection pour éviter des choix farfelus.

Ligne 215Ø: retour au programme principal.

Vous obtiendrez le caractère graphique des lignes 16Ø, 26Ø, 51Ø, 61Ø, 114Ø, 115Ø, 212Ø en tapant A après être passé en mode graphique G.



```
CHASE
    1 REM *********
    2 REM * CHASE *
    3 REM *********
    4 REM
    5 GO SUB 2000
   10 FOR p=1 TO n
   20 GO SUB 1000
   30 GO TO INT (RND*2+1)*100
  100 REM deplacement joueur 1
  110 LET b1=IN 65022: IF b1=255
THEN LET b1=a1
  120 LET x1=x1+(b1=247)-(b1=254)
  130 LET y1=y1+(b1=253)-(b1=251)
  140 IF ATTR (y1,x1)<>4 THEN GO
TO 600
  150 LET a1=b1
  160 PRINT AT y1,x1; INK 5;"■"
  200 REM deplacement joueur 2
  210 LET b2=IN 57342: IF b2=255
THEN LET b2=a2
  220 LET x2=x2+(b2=254)-(b2=247)
```

```
 230 LET y2=y2+(b2=251)-(b2=253)
 240 IF ATTR (y2,x2)<>4 THEN GO
TO 500
 250 LET a2=b2
 260 PRINT AT y2,x2; INK 2;"✿"
 270 GO TO 100
 500 LET s(1)=s(1)+1
 510 PRINT AT y2,x2; INK 4; FLAS
H 1;"✿": GO SUB 800
 520 GO TO 700
 600 LET s(2)=s(2)+1
 610 PRINT AT y1,x1; INK 4; FLAS
H 1;"✿": GO SUB 800
 700 PAUSE 4E4: PAUSE 4E4
 710 CLS : PRINT AT 1,3; INK 5;"
JOUEUR 1: ";s(1)
 720 PRINT AT 10,20; INK 2;"JOUE
UR 2: ";s(2)
 730 PAUSE 4E4: NEXT p
 740 INPUT "voulez vous rejouer
";r$
 750 IF r$(1)<>"n" THEN RUN
 760 STOP
 800 FOR i=1 TO 10
 810 FOR j=1 TO 10
 820 BEEP 0.005,j
 830 NEXT j
 840 NEXT i
 850 RETURN
1000 CLS
1010 LET x1=3: LET y1=10
1020 LET x2=28: LET y2=10
1030 LET a1=247: LET a2=247
1040 LET a$="████████████████████████████████
████████████████"
1050 PRINT AT 0,0; INK 6;a$;AT 2
1,0; INK 6;a$
1060 FOR i=1 TO 20
1070 PRINT AT i,0; INK 6;"█";AT
i,31; INK 6;"█"
1080 NEXT i
1090 FOR i=9 TO 0 STEP -1
1100 PRINT AT 10,16; INK 7;i
1110 BEEP 0.1,50
1120 NEXT i
1130 PRINT AT 10,16;" "
1140 PRINT AT y1,x1; INK 5;"✿"
1150 PRINT AT y2,x2; INK 2;"✿"
1160 RETURN
2000 BORDER 0: PAPER 0: INK 4: C
LS
2010 DIM s(2)
2020 FOR i=0 TO 7
2030 READ a
2040 POKE USR "A"+i,a
2050 NEXT i
2060 DATA 60,24,189,231,231,189,
24,60
```

```
2070 PRINT AT 4,3; INK 1;"
2080 PRINT AT 5,3; INK 2;"
2090 PRINT AT 6,3; INK 3;"
2100 PRINT AT 7,3; INK 4;"
2110 PRINT AT 8,3; INK 5;"
2120 FLASH 1: PRINT AT 15,9; INK
 5;"";AT 15,22; INK 2;"": FLAS
H 0
2130 INPUT "nombre de manches ?"
;n
2140 LET n=INT (ABS (n)): IF n<1
 THEN GO TO 2130
2150 RETURN
```

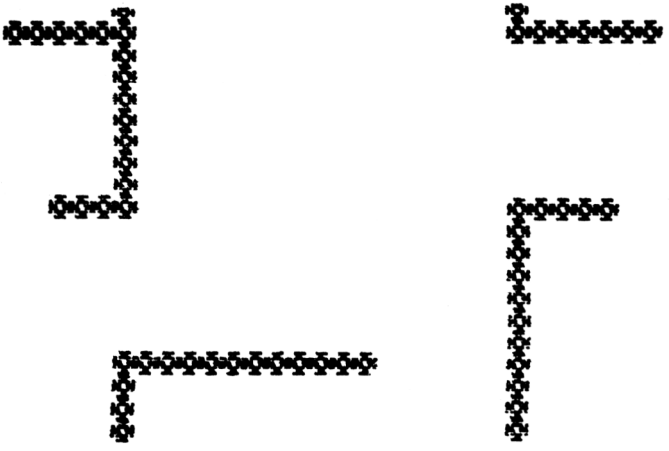
CHASE (version clavier). Terrain "sphérique"

Le programme que vous allez trouver ci-dessous est très semblable au programme précédent. Les règles du jeu sont identiques si ce n'est que, cette fois, le terrain n'est plus limité. Les véhicules qui sortent de l'écran réapparaissent du côté opposé au côté de leur disparition. Cela permet de tendre des pièges à son adversaire mais attention la tactique à employer n'est plus du tout la même que dans un terrain limité.

Structure du programme

Les lignes 12Ø, 13Ø, 22Ø et 23Ø du programme précédent sont modifiées pour permettre aux véhicules de sortir de l'écran pour réapparaître du côté opposé.

Les lignes 1Ø4Ø à 1Ø8Ø qui dessinaient les limites du terrain de jeu n'ont plus de raison d'être et disparaissent.

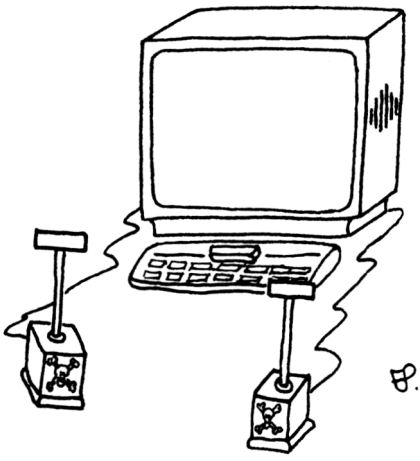


```
    1 REM *********
    2 REM * CHASE *
    3 REM *********
    4 REM
    5 GO SUB 2000
   10 FOR p=1 TO n
   20 GO SUB 1000
   30 GO TO INT (RND*2+1)*100
  100 REM deplacement joueur 1
  110 LET b1=IN 65022: IF b1=255
THEN LET b1=a1
  120 LET x1=x1+(b1=247)-(b1=254)
: IF x1=31 OR x1=0 THEN LET x1=A
BS (x1-30)
  130 LET y1=y1+(b1=253)-(b1=251)
: IF y1=21 OR y1=0 THEN LET y1=A
BS (y1-20)
  140 IF ATTR (y1,x1)<>4 THEN GO
TO 600
  150 LET a1=b1
  160 PRINT AT y1,x1; INK 5;"X"
  200 REM deplacement joueur 2
  210 LET b2=IN 57342: IF b2=255
THEN LET b2=a2
  220 LET x2=x2+(b2=254)-(b2=247)
: IF x2=31 OR x2=0 THEN LET x2=A
BS (x2-30)
  230 LET y2=y2+(b2=251)-(b2=253)
: IF y2=21 OR y2=0 THEN LET y2=A
BS (y2-20)
  240 IF ATTR (y2,x2)<>4 THEN GO
TO 500
  250 LET a2=b2
  260 PRINT AT y2,x2; INK 2;"X"
  270 GO TO 100
  500 LET s(1)=s(1)+1
```

```
 510 PRINT AT y2,x2; INK 4; FLAS
H 1;"✿": GO SUB 800
 520 GO TO 700
 600 LET s(2)=s(2)+1
 610 PRINT AT y1,x1; INK 4; FLAS
H 1;"✿": GO SUB 800
 700 PAUSE 4E4: PAUSE 4E4
 710 CLS : PRINT AT 1,3; INK 5;"
JOUEUR 1: ";s(1)
 720 PRINT AT 10,20; INK 2;"JOUE
UR 2: ";s(2)
 730 PAUSE 4E4: NEXT p
 740 INPUT "voulez vous rejouer
";r$
 750 IF r$(1)<>"n" THEN RUN
 760 STOP
 800 FOR i=1 TO 10
 810 FOR j=1 TO 10
 820 BEEP 0.005,j
 830 NEXT j
 840 NEXT i
 850 RETURN
1000 CLS
1010 LET x1=3: LET y1=10
1020 LET x2=28: LET y2=10
1030 LET a1=4: LET a2=32
1090 FOR i=9 TO 0 STEP -1
1100 PRINT AT 10,16; INK 7;i
1110 BEEP 0.1,50
1120 NEXT i
1130 PRINT AT 10,16;" "
1140 PRINT AT y1,x1; INK 5;"✿"
1150 PRINT AT y2,x2; INK 2;"✿"
1160 RETURN
2000 BORDER 0: PAPER 0: INK 4: C
LS
2010 DIM s(2)
2020 FOR i=0 TO 7
2030 READ a
2040 POKE USR "A"+i,a
2050 NEXT i
2060 DATA 60,24,189,231,231,189,
24,60
2070 PRINT AT 4,3; INK 1;"▀▀▀▀
▀ ▀   ▀▀▀▀  ▀▀▀▀  ▀▀▀▀"
2080 PRINT AT 5,3; INK 2;"▀
▀ ▀   ▀ ▀   ▀     ▀"
2090 PRINT AT 6,3; INK 3;"▀
▀▀▀▀  ▀▀▀▀  ▀▀▀▀  ▀▀"
2100 PRINT AT 7,3; INK 4;"▀
▀ ▀   ▀ ▀   ▀     ▀"
2110 PRINT AT 8,3; INK 5;"▀▀▀▀
▀ ▀   ▀ ▀   ▀▀▀▀  ▀"
2120 FLASH 1: PRINT AT 15,9; INK
 5;"✿";AT 15,22; INK 2;"✿": FLAS
H 0
2130 INPUT "nombre de manches ?"
;n
2140 LET n=INT (ABS (n)): IF n<1
 THEN GO TO 2130
2150 RETURN
```

3.3. CHASE (VERSION MANETTES DE JEU)

Ce jeu est identique à CHASE (version clavier) si ce n'est que les joueurs contrôlent les véhicules avec les manettes de jeu et non plus avec le clavier.



Structure du programme

Le programme est identique au programme CHASE clavier sauf:

Lignes 11Ø à 13Ø: lecture de la manette 1 et calcul de la nouvelle position du véhicule blanc.

Lignes 21Ø à 23Ø: lecture de la manette 2 et calcul de la nouvelle position du véhicule rouge.

Ligne 1Ø3Ø: initialisation des directions des deux véhicules.

```
  1 REM *********
  2 REM * CHASE *
  3 REM *********
  4 REM
  5 GO SUB 2000
 10 FOR p=1 TO n
 20 GO SUB 1000
 30 GO TO INT (RND*2+1)*100
100 REM deplacement joueur 1
110 LET p1=255-IN 63486: IF p1=
15 THEN LET p1=32
111 LET b1=p1+(255-IN 61438)
112 IF b1=0 THEN LET b1=a1
```

```
 120 LET x1=x1+(b1=4)-(b1=32)
 130 LET y1=y1+(b1=16)-(b1=8)
 140 IF ATTR (y1,x1)<>4 THEN GO
TO 600
 150 LET a1=b1
 160 PRINT AT y1,x1; INK 5;"⯐"
 200 REM deplacement joueur 2
 210 LET p2=255-IN 64510: IF p2=
16 THEN LET p2=32
 211 LET b2=p2+(255-IN 57342)
 212 IF b2=0 THEN LET b2=a2
 220 LET x2=x2+(b2=4)-(b2=32)
 230 LET y2=y2+(b2=16)-(b2=8)
 240 IF ATTR (y2,x2)<>4 THEN GO
TO 500
 250 LET a2=b2
 260 PRINT AT y2,x2; INK 2;"⯐"
 270 GO TO 100
 500 LET s(1)=s(1)+1
 510 PRINT AT y2,x2; INK 4; FLAS
H 1;"⯐": GO SUB 800
 520 GO TO 700
 600 LET s(2)=s(2)+1
 610 PRINT AT y1,x1; INK 4; FLAS
H 1;"⯐": GO SUB 800
 700 PAUSE 4E4: PAUSE 4E4
 710 CLS : PRINT AT 1,3; INK 5;"
JOUEUR 1: ";s(1)
 720 PRINT AT 10,20; INK 2;"JOUE
UR 2: ";s(2)
 730 PAUSE 4E4: NEXT p
 740 INPUT "voulez vous rejouer
";r$
 750 IF r$(1)<>"n" THEN RUN
 760 STOP
 800 FOR i=1 TO 10
 810 FOR j=1 TO 10
 820 BEEP 0.005,j
 830 NEXT j
 840 NEXT i
 850 RETURN
1000 CLS
1010 LET x1=3: LET y1=10
1020 LET x2=28: LET y2=10
1030 LET a1=4: LET a2=32
1040 LET a$="                "
1050 PRINT AT 0,0; INK 6;a$;AT 2
1,0; INK 6;a$
1060 FOR i=1 TO 20
1070 PRINT AT i,0; INK 6;"■";AT
i,31; INK 6;"■"
1080 NEXT i
1090 FOR i=9 TO 0 STEP -1
1100 PRINT AT 10,16; INK 7;i
1110 BEEP 0.1,50
1120 NEXT i
1130 PRINT AT 10,16;" "
1140 PRINT AT y1,x1; INK 5;"⯐"
```

```
1150 PRINT AT y2,x2; INK 2;"Ø"
1160 RETURN
2000 BORDER 0: PAPER 0: INK 4: C
LS
2010 DIM s(2)
2020 FOR i=0 TO 7
2030 READ a
2040 POKE USR "A"+i,a
2050 NEXT i
2060 DATA 60,24,189,231,231,189,
24,60
2070 PRINT AT 4,3; INK 1;"
2080 PRINT AT 5,3; INK 2;"
2090 PRINT AT 6,3; INK 3;"
2100 PRINT AT 7,3; INK 4;"
2110 PRINT AT 8,3; INK 5;"
2120 FLASH 1: PRINT AT 15,9; INK
 5;"Ø";AT 15,22; INK 2;"Ø": FLAS
H 0
2130 INPUT "nombre de manches ?"
;n
2140 LET n=INT (ABS (n)): IF n<1
 THEN GO TO 2130
2150 RETURN
```

CHASE (version manettes de jeu) terrain spérique.

De même que pour CHASE clavier, nous proposons une version "sphérique" du jeu.

Les modifications à apporter concernent les lignes 12Ø, 13Ø, 22Ø, 23Ø. Les lignes 1Ø4Ø à 1Ø8Ø disparaissent.

```
   1 REM *********
   2 REM * CHASE *
   3 REM *********
   4 REM
   5 GO SUB 2000
  10 FOR p=1 TO n
  20 GO SUB 1000
  30 GO TO INT (RND*2+1)*100
 100 REM deplacement joueur 1
 110 LET p1=255-IN 63486: IF p1=
16 THEN LET p1=32
 111 LET b1=p1+(255-IN 61438)
 112 IF b1=0 THEN LET b1=a1
 120 LET x1=x1+(b1=4)-(b1=32): I
F x1=31 OR x1=0 THEN LET x1=ABS
(x1-30)
```

```
 130 LET y1=y1+(b1=16)-(b1=8): I
F y1=21 OR y1=0 THEN LET y1=ABS
(y1-20)
 140 IF ATTR (y1,x1)<>4 THEN GO
TO 500
 150 LET a1=b1
 160 PRINT AT y1,x1; INK 5;"◆"
 200 REM deplacement joueur 2
 210 LET p2=255-IN 64510: IF p2=
16 THEN LET p2=32
 211 LET b2=p2+(255-IN 57342)
 212 IF b2=0 THEN LET b2=a2
 220 LET x2=x2+(b2=4)-(b2=32): I
F x2=31 OR x2=0 THEN LET x2=ABS
(x2-30)
 230 LET y2=y2+(b2=16)-(b2=8): I
F y2=21 OR y2=0 THEN LET y2=ABS
(y2-20)
 240 IF ATTR (y2,x2)<>4 THEN GO
TO 500
 250 LET a2=b2
 260 PRINT AT y2,x2; INK 2;"◆"
 270 GO TO 100
 500 LET s(1)=s(1)+1
 510 PRINT AT y2,x2; INK 4; FLAS
H 1;"◆": GO SUB 600
 520 GO TO 700
 600 LET s(2)=s(2)+1
 610 PRINT AT y1,x1; INK 4; FLAS
H 1;"◆": GO SUB 600
 700 PAUSE 4E4: PAUSE 4E4
 710 CLS : PRINT AT 1,3; INK 5;"
JOUEUR 1: ";s(1)
 720 PRINT AT 10,20; INK 2;"JOUE
UR 2: ";s(2)
 730 PAUSE 4E4: NEXT p
 740 INPUT "voulez vous rejouer
";r$
 750 IF r$(1)<>"n" THEN RUN
 760 STOP
 800 FOR i=1 TO 10
 810 FOR j=1 TO 10
 820 BEEP 0.005,j
 830 NEXT j
 840 NEXT i
 850 RETURN
1000 CLS
1010 LET x1=3: LET y1=10
1020 LET x2=28: LET y2=10
1030 LET a1=247: LET a2=247
1090 FOR i=9 TO 0 STEP -1
1100 PRINT AT 10,16; INK 7;i
1110 BEEP 0.1,50
1120 NEXT i
1130 PRINT AT 10,16;" "
1140 PRINT AT y1,x1; INK 5;"◆"
1150 PRINT AT y2,x2; INK 2;"◆"
1160 RETURN
2000 BORDER 0: PAPER 0: INK 4: C
LS
```

```
2010 DIM s(2)
2020 FOR i=0 TO 7
2030 READ a
2040 POKE USR "A"+i,a
2050 NEXT i
2060 DATA 60,24,189,231,231,189,
24,60
2070 PRINT AT 4,3; INK 1;"■■■
■ ■   ■■■   ■■■   ■■■"
2080 PRINT AT 5,3; INK 2;"■
■ ■   ■ ■   ■     ■"
2090 PRINT AT 6,3; INK 3;"■
■■■   ■■■   ■■■   ■"
2100 PRINT AT 7,3; INK 4;"■
■ ■   ■ ■     ■   ■"
2110 PRINT AT 8,3; INK 5;"■■■
■ ■   ■ ■   ■■■   ■■■"
2120 FLASH 1: PRINT AT 15,9; INK
 5;"🕷";AT 15,22; INK 2;"🕷": FLAS
H 0
2130 INPUT "nombre de manches ?"
;n
2140 LET n=INT (ABS (n)): IF n<1
 THEN GO TO 2130
2150 RETURN
```

3.4. CHENILLE

Une petite chenille se déplace sur l'écran du téléviseur à la recherche de nourriture. Ce qu'elle préfère par dessus tout, ce sont les chiffres qui surgissent autour d'elle, particulièrement les gros !! Malheureusement pour elle, la petite chenille a la vue très basse, et ses chances de passer sur un chiffre pour le manger sont bien faibles si quelqu'un ne vient pas à son secours pour la guider.

Vous aurez certainement à cœur d'aider cette chenille à se nourrir en la dirigeant à l'aide des touches 5, 6, 7 et 8 (ou bien avec la manette de jeu 1), et cela d'autant plus que votre score à ce jeu est égal à la somme des chiffres mangés par la chenille.

Chaque fois que la chenille avale un chiffre, elle dépose aussitôt derrière elle une traînée de petits cailloux dont le nombre dépend du chiffre "avalé".

Toute collision avec les bords du terrain ou bien avec l'un des cailloux déposés par la chenille est fatale à celle-ci. C'est toujours de cette façon que se termine le jeu.

Pour voir le score réalisé, pressez, en fin de partie, une touche quelconque du clavier ou bien le bouton rouge de la manette de jeu. Vous verrez alors apparaître le record actuel, la moyenne réalisé depuis le début du jeu et le score obtenu au cours de la dernière partie. Il suffit de presser une touche quelconque pour rejouer.

```
RECORD BATTU !!!

RECORD : 225
moyenne sur 5 parties : 91
SCORE : 225

pressez une touche pour rejouer
```

Structure du programme

- *Programme principal* : lignes 1 à 7Ø

Ligne 1Ø : appel du sous-programme de présentation.
Ligne 2Ø : appel du sous-programme d'initialisation.
Ligne 3Ø : appel du sous-programme de jeu.
Ligne 4Ø : appel du sous-programme de fin de partie.
Lignes 5Ø à 7Ø : proposition de rejouer.

- *Sous-programme de fin de partie* : lignes 1ØØØ à 1Ø8Ø

Ligne 1ØØØ : attente d'une pression de touche.
Ligne 1Ø1Ø : effacement de l'écran.

Ligne 1Ø2Ø: la variable N compte le nombre de parties jouées.

Ligne 1Ø3Ø: la variable TOTAL contient la somme des scores réalisés au cours des parties précédentes.

Ligne 1Ø35: calcul de la moyenne réalisée par le joueur depuis qu'il joue.

Ligne Ø4Ø: met à jour la variable RECORD si l'ancien record vient d'être battu.

Lignes 1Ø5Ø à 1Ø7Ø: affichage de record, score et moyenne.

Ligne 1Ø8Ø: retour au programme principal.

- *Sous-programme de jeu*: lignes 2ØØØ à 257Ø

Ligne 2ØØØ: affichage du score en haut de l'écran.

Ligne 2Ø1Ø: affichage de la petite chenille à sa nouvelle position.

Ligne 2Ø2Ø: lecture du clavier. Si aucune touche n'est pressée, la chenille continue d'avancer dans son ancienne direction.

Ligne 2Ø3Ø: les variables a$, ax et ay contiennent le sens de déplacement et les coordonnées de la chenille.

Lignes 2Ø4Ø et 2Ø5Ø: calcul des nouvelles coordonnées de la chenille en fonction de la touche pressée.

Ligne 2Ø6Ø: si la chenille n'a pas avancé, alors fin de la partie.

Ligne 2Ø7Ø: lecture du code du caractère présent à la nouvelle position de la chenille.

Ligne 2Ø8Ø: si aucun obstacle ne vient entraver la progression de la chenille, celle-ci poursuit sa route.

Ligne 2Ø9Ø: si la chenille heurte une étoile (code ASCII=42) ou un mur (code ASCII=128) alors fin de la partie.

Ligne 21ØØ: si la chenille passe sur un chiffre, le score (variable s) est augmenté de la valeur du chiffre "mangé". Le compteur de cailloux à déposer (variable e) est lui aussi augmenté de cette valeur.

Ligne 211Ø: affichage de la chenille sur sa nouvelle position.

Ligne 212Ø: la chenille dépose des cailloux derrière elle s'il y a lieu de le faire (e > Ø).

Ligne 213Ø: décide suivant la valeur prise par RND de placer un nouveau chiffre sur le terrain de jeu en appelant un sous-programme spécial.

Ligne 214Ø : retour au début de la boucle de jeu.

Ligne 25ØØ : affichage clignotant de la chenille qui vient de heurter soit un mur soit un caillou.

Ligne 251Ø : s'il restait des cailloux à déposer, on en pose un dernier...

Lignes 252Ø à 256Ø : routine de sonorisation.

Ligne 257Ø : retour au programme principal pour la fin de partie.

- *Sous-programme de placement d'un chiffre* : lignes 3ØØØ à 3Ø3Ø

Ligne 3ØØØ : choisit la position où va apparaître le chiffre.

Ligne 3Ø1Ø : si la position choisie par la ligne 3ØØØ n'est pas libre, un autre choix doit être fait.

Ligne 3Ø2Ø : apparition du chiffre.

Ligne 3Ø3Ø : retour au sous-programme de jeu.

- *Sous-programme d'initialisation* : lignes 4ØØØ à 413Ø

Ligne 4ØØØ : mise à zéro du score et du compteur de cailloux. La chenille partira, sauf contre-ordre du joueur, vers la droite.

Ligne 4Ø1Ø : initialisation de la position de départ de la chenille.

Lignes 4Ø2Ø à 4Ø6Ø : affichage du terrain de jeu.

Lignes 4Ø8Ø à 412Ø : compte à rebours.

Ligne 413Ø : retour au programme principal.

- *Sous-programme de présentation* : lignes 5ØØØ à 539Ø

Ligne 5ØØØ : l'écran et son bord seront de couleur noire ; l'encre des caractères sera verte. Effacement de l'écran.

Ligne 5ØØ5 : initialisation du générateur aléatoire.

Ligne 5Ø1Ø à 5Ø5Ø : définition du caractère graphique "A" qui représente la petite chenille.

Ligne 5Ø6Ø : la variable p$ est utilisée pour l'affichage du terrain de jeu.

Lignes 5Ø8Ø à 51ØØ : apparition de 11 chiffres sur l'écran.

Lignes 511Ø à 515Ø : apparition de 31 ensembles de deux étoiles.

Ligne 516Ø : initialisation de deux variables utilisées pour la présentation du jeu.

Lignes 517Ø à 532Ø: affichage en grosses lettres du nom du jeu.

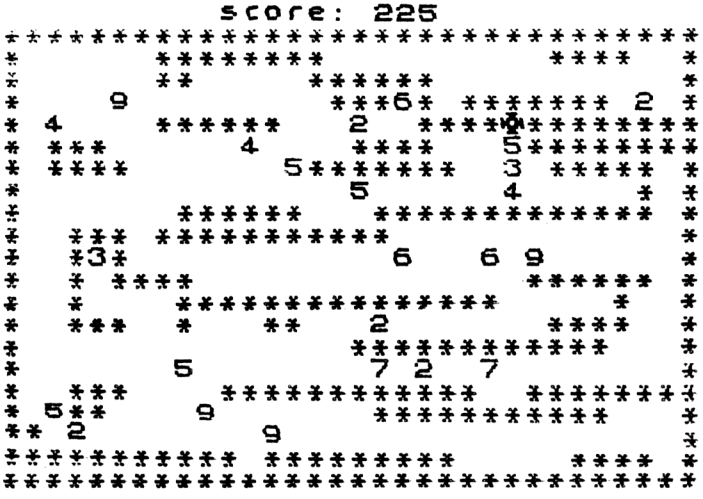
Ligne 533Ø: appel de la routine de sonorisation incluse dans le sous-programme de jeu.

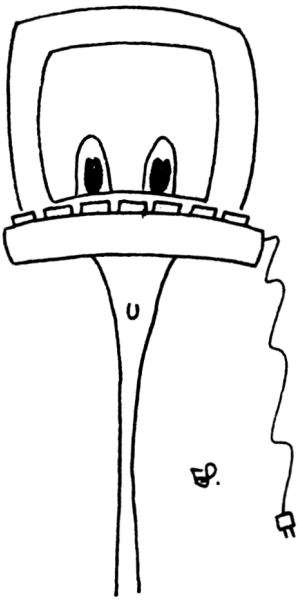
Lignes 534Ø et 535Ø: attente de la pression d'une touche pour continuer.

Ligne 536Ø: effacement de l'écran.

Lignes 537Ø et 538Ø: mise à zéro de la variable TOTAL qui comptabilise le nombre de points obtenus par le joueur depuis le début du jeu et du nombre de parties jouées (variable N) ainsi que le record (variable RECORD).

Ligne 539Ø: retour au programme principal.





```
   1 REM  ************
   2 REM  * CHENILLE *
   3 REM  ************
   4 REM
  10 GO SUB 5000
  20 GO SUB 4000
  30 GO SUB 2000
  40 GO SUB 1000
  50 PRINT AT 21,0;"pressez une
touche pour rejouer"
  60 PAUSE 0
  70 CLS : GO TO 20
1000 PAUSE 0: PAUSE 0
1010 CLS
1020 LET n=n+1
1030 LET total=total+s
1035 LET moyenne=INT (total/n)
1040 IF s>record THEN LET record
=s: PRINT AT 5,10; INK 7; FLASH
1;"  RECORD BATTU !!!": GO SUB 2
520
1050 PRINT AT 10,5; INK 4;"RECOR
D : ";record
1060 PRINT AT 14,5; INK 2;"SCORE
: ";s
1070 PRINT AT 12,5; INK 5;"moyen
ne sur ";n;" parties : ";moyenne
1080 RETURN
2000 PRINT AT 0,10; INK 7;"score
: ";s
2010 PRINT AT y,x;"A"
```

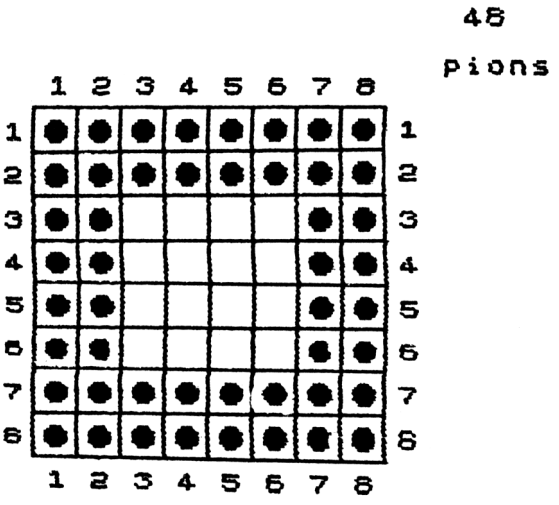
```
2020 LET b$=INKEY$: IF b$="" THE
N LET b$=a$
2030 LET a$=b$: LET ax=x: LET ay
=y
2040 LET x=x+(b$="8")-(b$="5")
2050 LET y=y+(b$="6")-(b$="7")
2060 IF x=ax AND y=ay THEN GO TO
 2500
2070 LET c=CODE (SCREEN$ (y,x))
2080 IF c=32 THEN GO TO 2110
2090 IF (c-42)*(c-128)=0 THEN GO
 TO 2500
2100 LET s=s+c-48: LET e=e+c-48
2110 PRINT AT y,x;"A";AT ay,ax;"
 "
2120 IF e>0 THEN PRINT AT ay,ax;
 INK 5;"*": LET e=e-1
2130 IF RND>0.9 THEN GO SUB 3000
2140 GO TO 2000
2500 PRINT AT y,x; INK 1; FLASH
1;"A";AT ay,ax; FLASH 0;" "
2510 IF e>0 THEN PRINT AT ay,ax;
 INK 5;"*"
2520 FOR i=1 TO 10
2530 FOR j=1 TO 10
2540 BEEP 0.005,j
2550 NEXT j
2560 NEXT i
2570 RETURN
3000 LET h=INT (RND*28+2): LET v
=INT (RND*20+1)
3010 IF SCREEN$ (v,h)<>" " THEN
GO TO 3000
3020 PRINT AT v,h; INK 7;INT (RN
D*8+2)
3030 RETURN
4000 LET s=0: LET e=0: LET a$="8
"
4010 LET x=10: LET y=10: LET ax=
x: LET ay=y
4030 PRINT AT 1,0; INK 2;p$;AT 2
1,0; INK 2;p$
4040 FOR i=1 TO 21
4050 PRINT AT i,0; INK 2;"*";AT
i,31; INK 2;"*"
4060 NEXT i
4080 FOR i=9 TO 0 STEP -1
4090 PRINT AT 10,16; INK 7;i
4100 BEEP 0.1,50
4110 NEXT i
4120 PRINT AT 10,16;" "
4130 RETURN
5000 BORDER 0: PAPER 0: INK 4: C
LS
5005 RANDOMIZE
5010 FOR i=0 TO 7
5020 READ a
5030 POKE USR "A"+i,a
5040 NEXT i
```

```
5050 DATA 60,24,189,231,231,189,
24,60
5060 LET p$="********************
************"
5080 FOR i=0 TO 10
5090 GO SUB 3000: BEEP 0.01,20
5100 NEXT i
5110 FOR i=0 TO 30
5120 LET h=INT (RND*28+2): LET v
=INT (RND*19+1)
5130 LET c=INT (RND*6+1)
5140 PRINT AT v,h; INK c;"**"
5150 NEXT i
5160 LET d$="AAAA": LET e$="A  A
"
5170 PRINT AT 2,4; INK 2;e$
5180 PRINT AT 3,4; INK 2;e$; INK
 3;d$
5190 PRINT AT 4,4; INK 2;d$; INK
 3;"A"
5200 PRINT AT 5,0; INK 1;d$; INK
 2;e$; INK 3;"AAA"
5210 PRINT AT 6,0; INK 1;"A   ";
 INK 2;e$; INK 3;"A   "; INK 4;e
$
5220 PRINT AT 7,0; INK 1;"A
  "; INK 3;d$; INK 4;"AA A"
5230 PRINT AT 8,0; INK 1;"A
  "; INK 4;"A AA"; INK 5;d$
5240 PRINT AT 9,0; INK 1;d$;AT 9
,12; INK 4;e$
5245 PRINT AT 9,16; INK 5;" AA
  "; INK 7;"A"
5250 PRINT AT 10,12; INK 4;e$; I
NK 5;" AA "; INK 6;"A   "; INK 7
;"A"
5260 PRINT AT 11,17; INK 5;"AA "
; INK 6;"A   "; INK 7;"A   "
6270 PRINT AT 12,16; INK 5;d$; I
NK 6;"A   "; INK 7;"A   "; INK 1
;d$
5280 PRINT AT 13,20; INK 6;"A
"; INK 7;d$; INK 1;"A"
5290 PRINT AT 14,20; INK 6;d$;AT
 14,28; INK 1;"AAA"
5300 PRINT AT 15,28; INK 1;"A"
5310 PRINT AT 16,28; INK 1;"A"
5320 PRINT AT 17,28; INK 1;d$
5330 GO SUB 2520
5340 PRINT AT 21,0;"pressez ENTE
R pour commencer"
5350 PAUSE 0
5360 CLS
5370 LET total=0: LET n=0
5380 LET record=0
5390 RETURN
```

3.5. DAMIER SOLITAIRE

DAMIER SOLITAIRE est un jeu de réflexion fort semblable dans son principe au célèbre jeu du solitaire. Le but de ce jeu est en effet d'arriver à ne garder sur le damier qu'un seul pion apèrs avoir éliminé, grâce à des prises successives, tous les autres pions.

Au début du jeu, quarante huit pions sont disposés sur un damier 8 × 8 dans la position suivante :



Les prises se font comme dans le jeu de dames, c'est-à-dire en diagonale. La case située derrière le pion que l'on désire prendre doit bien sûr être libre.

Le ZX Spectrum vous demande le mouvement que vous désirez effectuer. Entrez alors un nombre de quatre chiffres, les deux premiers chiffres étant la ligne et la colonne du pion de départ, les deux chiffres suivant représentant la ligne et la colonne de la case d'arrivée. Pressez ENTER pour valider votre réponse.

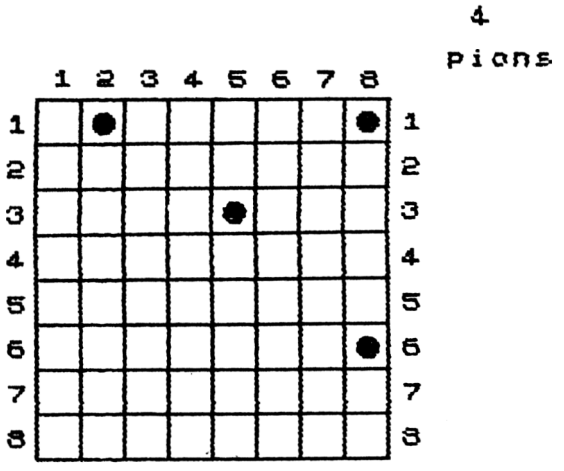
Exemple : 1133 "prend" le pion situé en 22 et place le pion initialement en 11 dans la case 33.

Le ZX Spectrum détecte toute erreur ou tentative de "fraude" qui conduit à un coup interdit ou impossible... Un autre mouvement vous est alors demandé.

Vous pouvez interrompre la partie à tout moment en pressant simplement la touche ENTER en réponse à la question MOUVEMENT ? Cela est particulièrement utile quand, en fin de partie, plus aucune prise n'est possible. Le ZX Spectrum vous proposera alors de rejouer.

Pour rejouer, pressez les touches O puis ENTER.

DAMIER SOLITAIRE est un jeu très difficile. Enlever entre 4Ø et 45 pions est déjà une belle performance, en enlever 46 ou 47 est un exploit ! Alors bon courage...



Structure du programme

- *Programme principal* : lignes 1 à 23Ø

Ligne 5 : change le bruit produit par le clavier.

Ligne 7 : l'écran et son bord seront de couleur noire ; l'encre des caractères sera verte.

Ligne 8 : appel du sous-programme de définition du caractère graphique "A" qui représentera un pion.

Ligne 1Ø : déclaration des tableaux de travail ; a contiendra le mouvement en cours et t représente le damier.

Lignes 2Ø à 55 : les cases ne contenant pas de pions sont repérées dans le tableau t par des 1.

Ligne 57 : appel du sous-programme de présentation.

Ligne 6Ø : appel du sous-programme d'affichage du damier et des pions.

Ligne 75 : demande au joueur d'entrer le mouvement désiré.

Ligne 8Ø : si le joueur presse simplement ENTER, la partie est finie.

Ligne 85 : si le joueur entre un mouvement comportant moins de quatre chiffres, sa proposition est refusée.

Lignes 9Ø à 11Ø : analyse du mouvement ; a(1) et a(2) contiennent les coordonnées du pion qui va sauter ; a(3) et a(4) contiennent les coordonnées de la case vide qui va recevoir le pion.

Lignes 14Ø à 16Ø : vérification que les cases de départ et d'arrivée appartiennent bien au damier.

Ligne 17Ø : teste si la case de départ contient bien un pion et si la case d'arrivée est bien vide.

Ligne 18Ø : teste la validité de la prise (prise en diagonale comme au jeu de dames).

Ligne 19Ø : teste si la case par-dessus laquelle le pion saute, contient bien un pion.

Ligne 2ØØ : enlève le pion "mangé" et libère la case de départ ; met un pion dans la case d'arrivée.

Ligne 21Ø : le nombre de pions restants sur le damier est diminué d'une unité. Appel du sous-programme d'affichage des pions.

Ligne 22Ø : teste une éventuelle victoire...

Ligne 23Ø : retour au début de la boucle de jeu.

- *Routines utilitaires*

Lignes 5ØØ à 56Ø : cas d'un mouvement interdit.

Lignes 2ØØØ à 2Ø2Ø : proposition de rejouer.

- *Sous-programme d'affichage des pions* : lignes 1ØØØ à 1Ø2Ø

Ligne 1ØØØ : enlève le pion de départ et le pion mangé ; met un pion dans la case d'arrivée.

Ligne 1Ø1Ø : affiche le nombre de pions restants.

Ligne 1Ø2Ø : retour au programme principal.

- *Sous-programme de redéfinition d'un caractère* : lignes 8ØØØ à 8Ø5Ø

Lignes 8ØØØ à 8Ø5Ø : redéfinition du caractère graphique "A" qui doit représenter un pion.

- *Sous-programme d'affichage du damier et les pions* : lignes 9ØØØ à 918Ø

Lignes 9ØØØ à 9Ø5Ø : affichage du damier.

Lignes 9Ø55 à 91ØØ : affichage des numéros des lignes et de colonnes du damier.

Lignes 913Ø à 917Ø : affichage des pions dans leur position initiale.

Ligne 918Ø : retour au programme principal ou au programme de présentation suivant le cas.

- *Sous-programme de présentation* : lignes 95ØØ à 957Ø

Ligne 95ØØ : la couleur du damier est choisie par le contenu de la variable a, celle des pions par la variable b.

Ligne 951Ø : effacement de l'écran. Appel du sous-programme d'affichage du damier et des pions.

Ligne 952Ø : affichage du nom du jeu.

Lignes 953Ø à 9534 : disparition des pions.

Lignes 9535 et 9536 : choix de la position du dernier pion.

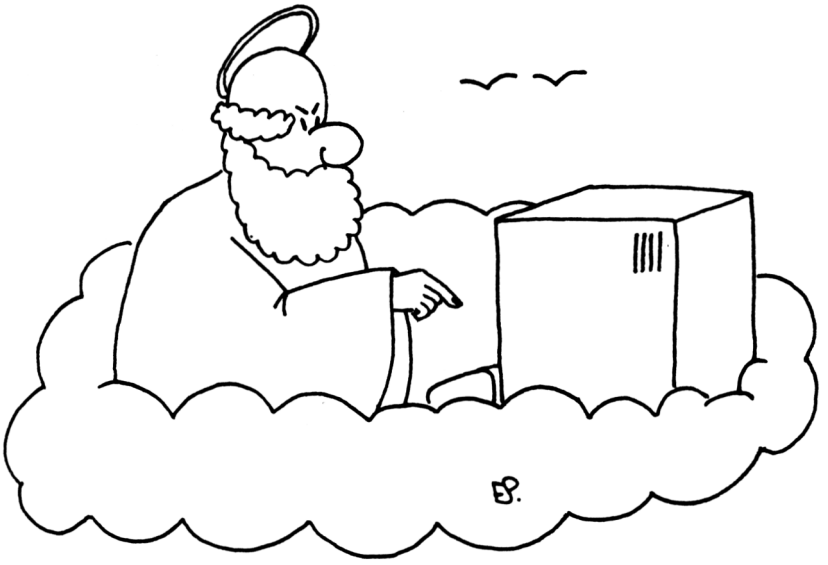
Ligne 9537 : affichage clignotant du dernier pion.

Lignes 954Ø à 9545 : routine de sonorisation.

Ligne 955Ø : effacement de l'écran.

Ligne 956Ø : choix des couleurs du damier et des pions.

Ligne 957Ø : retour au programme principal.



```
    1 REM **********************
    2 REM * DAMIER SOLITAIRE *
    3 REM **********************
    4 REM
    5 POKE 23609,100
    7 BORDER 0: PAPER 0: INK 4: C
LS
    8 GO SUB 8000
   10 DIM a(4): DIM t(8,8): LET n
=48
   20 FOR i=3 TO 6
   30 FOR j=3 TO 6
   40 LET t(i,j)=1
   50 NEXT j
   55 NEXT i
   57 GO SUB 9500
   60 GO SUB 9000
   65 PRINT AT 0,28;"48";AT 2,27;
"pions"
   70 PRINT AT 0,0;"
     "
   75 INPUT "mouvement ?",m$
   80 IF m$="" THEN GO TO 2000
   85 IF LEN m$<>4 THEN GO TO 75
   90 FOR i=1 TO 4
   96 LET a(i)=VAL m$(i)
  100 IF a(i)=0 THEN GO TO 75
  110 NEXT i
  140 FOR i=1 TO 4
  150 IF (a(i)-1)*(a(i)-8)>0 THEN
 GO TO 500
  160 NEXT j
```

```
 170 IF t(a(1),a(2))<>0 OR t(a(3
),a(4))=0 THEN GO TO 500
 180 IF ABS (a(1)-a(3))<>2 OR AB
S (a(2)-a(4))<>2 THEN GO TO 500
 190 IF t(0.5*(a(1)+a(3)),0.5*(a
(2)+a(4)))<>0 THEN GO TO 500
 200 LET t(0.5*(a(1)+a(3)),0.5*(
a(2)+a(4)))=1: LET t(a(1),a(2))=
1: LET t(a(3),a(4))=0
 210 LET h=h-1: GO SUB 1000
 220 IF h=1 THEN GO TO 2000
 230 GO TO 70
 500 PRINT AT 0,0;"mouvement int
erdit"
 510 FOR k=1 TO 10
 520 BEEP 0.01,10
 530 BEEP 0.01,30
 540 BEEP 0.01,20
 550 NEXT k
 560 GO TO 70
1000 PRINT AT a(1)*2+3,a(2)*2+7;
" ";AT a(3)*2+3,a(4)*2+7; INK b;
"●";AT a(1)+a(3)+3,a(2)+a(4)+7;
 "
1010 PRINT AT 0,28;"  ";AT 0,28;
h
1020 RETURN
2000 INPUT "voulez vous rejouer
?",r$: IF r$="" THEN GO TO 2000
2010 IF r$(1)<>"o" THEN STOP
2020 RUN
8000 FOR i=0 TO 7
8010 READ a
8020 POKE USR "A"+i,a
8030 NEXT i
8040 DATA 60,126,255,255,255,255
,126,60
8050 RETURN
9000 FOR i=0 TO 8
9010 PLOT 67+16*i,140: DRAW INK
a,0,-128
9040 PLOT 67,140-16*i: DRAW INK
a,128,0
9050 NEXT i
9055 LET q=0
9060 FOR i=1 TO 8
9070 PRINT AT 3,9+q;i;AT 21,9+q;
i
9080 PRINT AT 5+q,7;i;AT 5+q,25;
i
9090 LET q=q+2
9100 NEXT i
9130 FOR i=1 TO 8
9140 FOR j=1 TO 8
9150 IF t(i,j)=0 THEN PRINT AT 2
*i+6,2*j+7; INK b;"●": BEEP 0.01
,10
9160 NEXT j
9170 NEXT i
9180 RETURN
9500 LET a=2: LET b=5
```

```
9510 CLS : GO SUB 9000
9520 PRINT AT 0,9; FLASH 1; BRIG
HT 1;"DAMIER SOLITAIRE"
9530 FOR i=1 TO 8
9531 FOR j=1 TO 8
9532 PRINT AT 2*i+3,2*j+7;" ": B
EEP .01,30
9533 NEXT j
9534 NEXT i
9535 LET i=INT (RND*8+1)
9536 LET j=INT (RND*8+1)
9537 PRINT AT 2*i+3,2*j+7; INK 5
; FLASH 1; BRIGHT 1;"●"
9540 FOR i=1 TO 20
9542 BEEP .01,20
9544 BEEP .01,10
9545 NEXT i
9550 CLS
9560 LET a=2: LET b=5
9570 RETURN
```

3.6. EXPLORATION SPATIALE (48 K)

```
EXPLORATION SPATIALE
********************

     Alors que vous etiez en
mission, votre vaisseau spatial
est tombe en panne, son atompile
etant epuisee.

     Apres un atterrissage en
catastrophe, vous etes perdu sur
une planete etrangere.

     Il vous faut donc trouver
au plus vite une atompile avant
de rejoindre la navette pour
quitter cette planete maudite.
```

La situation de départ est :

```
Vous etes dans une foret.
Des sentiers partent vers le
nord et le sud.

Je vois
un(e) torche
un(e) corde
un(e) radio
```

Votre but est donc de trouver une atompile, retourner à la navette et décoller. Pour ce faire, vous disposez de commandes pour vous déplacer et pour agir sur les objets que vous découvrirez au cours de vos déplacements (une atompile par exemple...).

Les commandes sont les suivantes :

COMMANDES

n	nord	s	sud
o	ouest	e	est
monte	descends	prends	pose
revoir	regarde	allume	donne
ouvre	debloque	nage	mange
brise	lit	decolle	help
fin	inventaire		

L'interpréteur de commandes accepte les majuscules et les minuscules. La syntaxe d'une commande est : verbe <un espace au moins> nom.

Exemples :

monte

prends^radio

pose^^touche ^ représente un espace.

REMARQUES :

- La commande "help" peut avoir des effets surprenants.
- La commande "revoir" redonne la description de la position courante.
- La commande "regarde" peut être utile pour voir des objets cachés.

Vous trouverez beaucoup d'objets sur votre route mais tous ne sont pas indispensables. Vous devrez donc faire de bons choix sachant qu'il vous est interdit de porter plus de quatre objets à la fois.

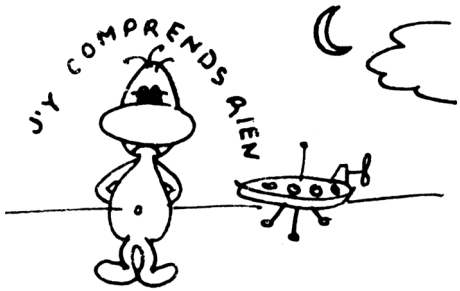
Par exemple, "prendre torche"peut être judicieux si vous voulez ensuite demander "allume torche" mais "prendre radio" pour faire "allume radio" n'est pas forcément très intéressant.

Un papier, bien caché, vous dira où trouver la réserve d'atompiles, saurez-vous le découvrir ?

La commande "inventaire" permet de vérifier quels sont les objets que vous transportez. Peut être serez-vous obligé de prendre certains objets puis de les reposer pour les reprendre ensuite ?

EXPLORATION SPATIALE est un jeu très difficile car vous ne connaissez ni la carte, ni toutes les opérations nécessaires pour retrouver l'atompile avant de rejoindre la navette et demander "décolle". De nombreux pièges vous guettent (marécages, précipices, guerriers, ...) et le temps joue contre vous : si vous ne trouvez pas de nourriture, vous mourrez de faim !

C'est volontairement que je ne donne pas plus de détails sur la carte, les commandes, les objets et les pièges car tout l'intérêt du jeu réside dans les tentatives, souvent infructueuses, que vous serez amené à faire pour reconnaître le paysage, choisir les bons objets, éviter les pièges. Vous ne gagnerez certainement pas du premier coup et j'espère même, dans votre propre intérêt, que vous aurez beaucoup de mal à gagner !



Structure du programme

- *Programme principal* : lignes 1 à 15Ø

Ligne 5Ø : appel du sous-programme de présentation.

Ligne 8Ø : début du jeu.

Lignes 1ØØ à 15Ø : fin de partie classique.

- *Affichage position et objets* : lignes 5ØØ à 555

Lignes 5ØØ à 515 : mise à zéro des pointeurs de déplacements et de la chaîne p$ de description.

Lignes 52Ø et 525 : lecture et affichage de la description du paysage dans lequel évolue le joueur.

Lignes 53Ø à 555 : affichage des objets qui se trouvent à cette position.

- *Interpréteur de commandes* : lignes 57Ø à 63Ø

Lignes 57Ø à 573 : calcul du temps écoulé et tests.

Ligne 574 : entrée d'une ligne de commande.

Lignes 575 à 585 : conversion majuscule-minuscule.

Lignes 595 à 625 : décodage de la commande et décomposition en deux morceaux (a$ et b$).

Ligne 63Ø : affichage de la commande demandée.

- *Exécution des commandes* : lignes 64Ø à 75Ø

Batterie de tests et renvoi à la commande concernée.

- *Sous-programmes de description du paysage* : lignes 1ØØØ à 15ØØ

Chaque emplacement est décrit par une variable p$ et des pointeurs qui indiquent les emplacements adjacents (nord, sud, est, ouest).

- *Commandes* : lignes 2ØØØ à 889Ø

Lignes 2ØØØ à 2Ø8Ø : prendre.

Lignes 25ØØ à 256Ø : monter.

Lignes 27ØØ à 273Ø : descendre.

Lignes 3ØØØ à 3Ø7Ø : poser.

Lignes 4ØØØ à 4Ø2Ø : regarder.

Lignes 45ØØ à 454Ø : éclairer.

Lignes 5ØØØ à 5Ø8Ø : ouvrir.

Lignes 55ØØ à 557Ø : débloquer.

Lignes 6ØØØ à 6Ø2Ø : nager.

Lignes 65ØØ à 657Ø : briser.

Lignes 7ØØØ à 7Ø8Ø : donner.

Lignes 72ØØ à 726Ø : manger.

Lignes 75ØØ à 755Ø : lire.

Lignes 8ØØØ à 8Ø8Ø : vérifier.

Lignes 85ØØ à 852Ø : décoller.

Lignes 87ØØ à 876Ø : fin.

Lignes 88ØØ à 889Ø : help.

- *Sous-programmes de présentation* : lignes 9ØØØ à 957Ø

Ligne 9Ø1Ø : déclaration du tableau p des positions occupées par les objets (Ø si l'objet est transporté par le joueur) et du tableau O$ des objets.

Ligne 9Ø2Ø : définition de la fonction t qui donne le temps écoulé depuis la dernière mise à zéro de l'horloge interne du ZX Spectrum.

Ligne 9Ø3Ø : mise à zéro du compteur c d'objets transportés.

Lignes 9Ø35 et 9Ø37 : mise à zéro de drapeaux.

Ligne 9Ø4Ø : initialisation position de départ.

Lignes 9Ø5Ø à 9Ø9Ø : paramétrage de la position des commandes dans le programme. Vous pouvez ainsi rajouter très facilement de nouvelles commandes ou modifier et déplacer celles qui existent déjà.

Lignes 91ØØ à 917Ø : dessin de présentation utilisant une particularité de la fonction DRAW x,y,z.

Lignes 918Ø à 923Ø : routine de sonorisation.

Lignes 95ØØ à 954Ø : initialisation des tableaux p et O$.

Ligne 955Ø : appel du sous-programme d'affichage de la règle du jeu.

Ligne 956Ø : mise à zéro de l'horloge.

Ligne 957Ø : retour au programme principal.

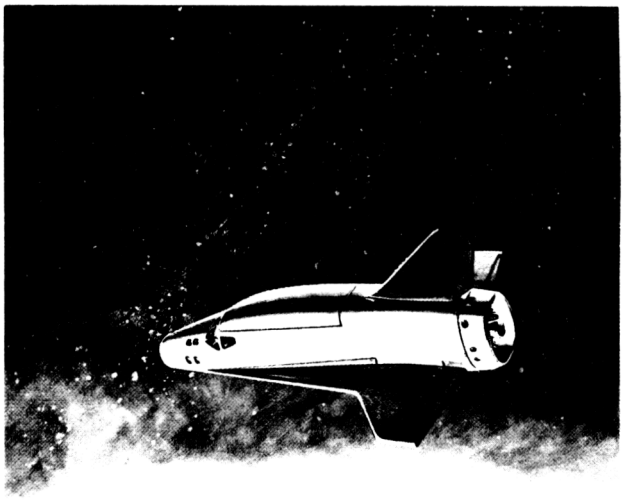
- *Sous-programme d'affichage de la règle du jeu* : lignes 97ØØ à 994Ø

Lignes 97ØØ à 978Ø : affichage de la règle du jeu.

Lignes 98ØØ à 992Ø : affichage des commandes.

Ligne 994Ø : retour soit au programme principal soit à la commande "help".

La structure de chacune des commandes n'est pas explicitée afin que vous découvriez en jouant comment les commandes agissent.



```
    1 REM ************************
    2 REM *EXPLORATION SPATIALE*
    3 REM ************************
    4 REM
    5 REM -------------------------
   10 REM © Alain PERBOST et les
          Editions EYROLLES
   20 REM -------------------------
   30 REM
   40 REM initialisations
   50 GO SUB 9000
   60 REM
   70 REM jeu
   80 GO SUB 500
   90 REM
  100 REM fin dela partie
  110 CLS
  120 PRINT AT 10,10;"Au revoir..
"
  130 INPUT "Voulez-vous rejouer
?"; LINE r$
  140 IF r$="" OR r$(1)="o" THEN
RUN
  150 STOP
  497 REM
  498 REM boucle de jeu
```

```
 499 REM
 500 LET nord=0: LET sud=0
 505 LET ouest=0: LET est=0
 510 LET monte=0: LET descendre=
0
 515 LET p$=""
 520 GO SUB position
 525 CLS : PRINT p$
 530 LET trouve=0
 535 FOR n=1 TO 12
 540 IF p(n)<>position THEN NEXT
 n: GO TO 560
 545 IF NOT trouve THEN PRINT :
PRINT "Je vois ": LET trouve=1
 550 PRINT "un(e) ";o$(n)
 555 NEXT n
 560 IF a$="regarde" AND NOT tro
uve THEN PRINT "Je ne vois rien
de particulier"
 565 IF position=1120 AND NOT p(
1) AND p(6) THEN PRINT "Vous ave
z glisse et, sans corde pour vou
s rattraper, vous etes  tombe da
ns un grand precipice...": GO TO
 fin
 570 LET temps=FN t()-473
 572 IF temps>100 AND temps<=300
 THEN PRINT INK 2;"J'ai faim !!!
": GO TO 574
 573 IF temps>320 THEN PRINT "Vo
us etes mort de faim...": GO TO
fin
 574 BEEP 0.01,20: INPUT INK 2;"
Que dois-je faire ?", LINE c$: I
F c$="" THEN GO TO 570
 575 FOR n=1 TO LEN c$
 580 IF c$(n)>="A" AND c$(n)<="Z
" THEN LET c$(n)=CHR$ (CODE c$(n
)+32)
 585 NEXT n
 590 IF (position=1450 OR positi
on=1480) AND RND>0.95 THEN PRINT
 "Le guerrier,irrite par votre c
omportement,vient de vous tuer":
 GO TO fin
 595 IF c$(1)=" " AND LEN c$>1 T
HEN LET c$=c$(2 TO ): GO TO 595
 600 LET blanc=0: LET a$="": LET
 b$=""
 605 FOR n=1 TO LEN c$
 610 IF c$(n)=" " AND NOT blanc
THEN LET blanc=1: GO TO 625
 615 IF NOT blanc THEN LET a$=a$
+c$(n)
 620 IF blanc THEN LET b$=b$+c$(
n)
 625 NEXT n
 630 PRINT INK 1;a$;" ";b$
 635 REM
 640 REM execution commande
```

```
  642 REM
  644 IF a$="" THEN GO TO 570
  645 IF a$="revoir" THEN GO TO 5
25
  650 IF (a$="n" OR a$="nord") AN
D nord THEN LET position=nord: G
O TO 500
  655 IF (a$="s" OR a$="sud") AND
 sud THEN LET position=sud: GO T
O 500
  660 IF (a$="o" OR a$="ouest") A
ND ouest THEN LET position=ouest
: GO TO 500
  665 IF (a$="e" OR a$="est") AND
 est THEN LET position=est: GO T
O 500
  670 IF a$="descends" THEN GO TO
 descente
  675 IF a$="monte" THEN GO TO mo
nter
  680 IF a$="prends" THEN GO TO p
rendre
  685 IF a$="pose" THEN GO TO pos
er
  690 IF a$="regarde" THEN GO TO
regarder
  695 IF a$="allume" THEN GO TO e
clairer
  700 IF a$="ouvre" THEN GO TO ou
vrir
  705 IF a$="debloque" THEN GO TO
 debloquer
  710 IF a$="nage" THEN GO TO nag
er
  715 IF a$="brise" THEN GO TO br
iser
  720 IF a$="donne" THEN GO TO do
nner
  722 IF a$="mange" THEN GO TO ma
nger
  725 IF a$="lit" THEN GO TO lire
  730 IF a$="inventaire" THEN GO
TO verifier
  735 IF a$="decolle" THEN GO TO
decoller
  740 IF a$="help" THEN LET x=RND
: GO TO help
  745 IF a$="fin" THEN GO TO fin
  750 PRINT "Je ne peux pas !!!":
 GO TO 570
  997 REM
  998 REM description du paysage
  999 REM
1000 LET p$="Vous etes dans une
foret.        Des sentiers parten
t vers le     nord et le sud."
1010 LET nord=1030: LET sud=1060
1020 RETURN
```

```
1030 LET p$="Vous etes dans un m
assif          d'arbustes. Le lac
est a l'est. La foret est au sud
 et une        prairie se trouve a
u nord."
1040 LET nord=1090: LET sud=1000
1050 RETURN
1060 LET p$="Vous etes dans une
foret. Des     sentiers partent ve
rs le nord et l'est. Une clairier
e se trouve al'ouest."
1070 LET nord=1000: LET est=1120
: LET ouest=1150
1080 RETURN
1090 LET p$="Vous etes dans une
prairie. Le    lac est au sud-est.
 Au nord se    trouve un massif d'
arbustes et al'est une grande to
ur blanche."
1099 REM
1100 LET est=1360: LET sud=1030
1110 RETURN
1120 LET p$="Vous etes dans la f
oret. Il y a un sentier dans la
direction est-ouest."
1130 LET ouest=1060: LET est=118
0
1140 RETURN
1150 LET p$="Vous etes dans une
clairiere. Aucentre il y a la na
vette           spatiale. Une sorti
e est visiblea l'est."
1160 LET est=1060
1170 RETURN
1180 LET p$="Vous etes dans une
foret. Grace a une eclaircie ent
re les arbresau nord, vous pouve
z apercevoir un lac. Le sentierv
a est-ouest. Il y a une petite c
averne au sud"
1190 LET est=1240: LET ouest=112
0: LET sud=1210
1200 RETURN
1210 LET p$="Vous etes dans une
petite          caverne. La lumiere
 du jour est visible au nord. Un
e porte se     trouve a l'est. Une
 ouverture     est visible au sud.
"
1220 LET sud=1270: LET nord=1180
1230 RETURN
1240 LET p$="Vous etes au bord d
'un lac. Il ya une ile au milieu
 du lac. La  foret est a l'est.
Au nord il y a des eboulis de ro
chers."
1250 LET nord=1300: LET ouest=11
80
1260 RETURN
```

```
1270 LET p$="Vous etes dans une
grande        caverne. Dans un co
in il y a un squelette et un cof
fre d'acier. La seule sortie est
 au nord."
1280 LET nord=1210
1290 RETURN
1300 LET p$="Vous etes dans un e
boulis de     rochers. Le lac est
 a l'ouest etil y a un marais au
 nord. Les    sorties possibles s
ont au nord   et au sud."
1310 LET nord=1330: LET sud=1240
1320 RETURN
1330 CLS : PRINT "Vous etes tomb
e dans un marecage": GO TO fin
1360 LET p$="Vous etes au pied d
'une grande   tour blanche. Aucun
e porte n'estvisible. A l'ouest
se trouve uneprairie. Le marais
est a l'est."
1370 LET ouest=1090: LET est=133
0: LET monte=1390
1380 RETURN
1390 LET p$="Vous etes au sommet
 de la tour. Au centre de la pla
teforme il y a un coffret en ver
re qui        contient une cle."
1400 LET descendre=1360
1410 RETURN
1420 LET p$="Je n'y vois rien !!
"
1430 LET monte=1210
1440 RETURN
1450 LET p$="Vous etes dans un s
outerrain.    Au nord il y a un e
scalier gardepar un guerrier. Un
e petite      piece est visible a
u sud."
1460 LET sud=1420
1470 RETURN
1480 LET p$="Vous etes sur l'ile
 au milieu dulac. Un escalier  q
ui descend aufond d'un grand tro
u est visible"
1490 LET descendre=1450
1500 RETURN
2000 REM prendre
2001 REM
2010 IF c>=4 THEN PRINT "Je ne p
eux pas porter davantage d'objet
s.": GO TO 570
2020 IF b$="" THEN INPUT " Prend
re quoi ?"; LINE b$: GO TO 2020
2030 LET trouve=0
2040 FOR n=1 TO 12
2050 IF o$(n)=(b$+"           ") (
 TO 10) AND p(n)=position THEN P
RINT "OK !!": LET trouve=1: LET
p(n)=0: LET c=c+1
```

```
2060 NEXT n
2070 IF NOT trouve THEN GO TO 75
0
2080 GO TO 570
2499 REM
2500 REM monter
2501 REM
2510 IF position=1450 THEN GO TO
 2560
2520 IF position<>1360 AND monte
 THEN LET position=monte: GO TO
500
2530 IF p(3) THEN PRINT "Il est
impossible de monter !!!": GO TO
 570
2540 IF monte THEN LET position=
monte: GO TO 500
2550 PRINT "Je ne peux pas monte
r !!": GO TO 570
2560 IF NOT tresor THEN PRINT "L
e guerrier demande un present.":
 GO TO 570
2570 LET position=1480: GO TO 50
0
2699 REM
2700 REM descente
2701 REM
2710 IF (position=1210 OR positi
on=1480) AND descendre THEN LET
position=descendre: GO TO 500
2720 IF descendre AND  NOT p(3)
THEN LET position=descendre: GO
TO 500
2730 GO TO 750
2999 REM
3000 REM poser
3001 REM
3005 IF position=1420 THEN GO TO
 750
3010 LET trouve=0
3020 FOR n=1 TO 12
3030 IF o$(n)=(b$+"          ")( T
O 10)  AND p(n)=0 THEN PRINT "OK
!!": LET trouve=1: LET p(n)=posi
tion: LET c=c-1
3040 IF b$="" THEN INPUT " Poser
 quoi ?"; LINE b$: GO TO 3040
3050 NEXT n
3060 IF NOT trouve THEN PRINT "J
e n'ai pas de ";b$;" !!!"
3070 GO TO 570
3999 REM
4000 REM regarder
4001 REM
4010 GO TO 530
4499 REM
4500 REM eclairer
4501 REM
4510 IF b$="" THEN INPUT "Allume
r quoi "; LINE b$: GO TO 4510
```

```
4520 IF b$<>"torche" THEN GO TO
750
4525 IF p(5) THEN GO TO 750
4530 IF position=1420 THEN LET p
$="Vous etes dans un petite piec
e. Il y a un escalier a l'est et
 unpassage sombre au nord.": LET
 nord=1450: GO TO 525
4540 PRINT "Ca ne sert pas a gra
nd chose...": GO TO 570
4999 REM
5000 REM ouvrir
5001 REM
5010 IF b$="" THEN INPUT "Ouvrir
 quoi ?"; LINE b$: GO TO 5010
5020 IF b$="porte" AND position<
>1210 THEN PRINT "Quelle porte ?
": GO TO 570
5030 IF b$="porte" AND NOT deblo
que THEN PRINT "La porte est blo
quee !!!": GO TO 570
5040 IF b$="porte" THEN PRINT "I
l y a un escalier qui descend.":
 LET descendre=1420: GO TO 570
5050 IF b$="coffre" AND position
<>1270 THEN PRINT "Je ne vois pa
s de coffre !!": GO TO 570
5060 IF b$="coffre" AND NOT ouve
rt THEN PRINT "Le coffre est fer
me a cle !!": GO TO 570
5070 IF b$="coffre" THEN PRINT "
D'accord !": LET p(4)=position:
LET p(11)=position: GO TO 570
5080 GO TO 750
5499 REM
5500 REM debloquer
5501 REM
5510 IF b$="" THEN INPUT "Debloq
uer quoi ?"; LINE b$: GO TO 5510
5520 IF b$="porte" AND position<
>1210 THEN PRINT "Quelle porte ?
": GO TO 570
5530 IF p(7) THEN PRINT "C'est i
mpossible sans la cle !!": GO TO
 570
5540 IF b$="porte" THEN LET debl
oque=1: PRINT "Click !!!": BEEP
0.1,10: GO TO 570
5550 IF b$="coffre" AND position
<>1270 THEN PRINT "Quel coffre ?
": GO TO 570
5560 IF b$="coffre" THEN LET ouv
ert=1: PRINT "Click !!!": BEEP 0
.1,5: GO TO 570
5570 GO TO 750
5999 REM
6000 REM nager
6001 REM
```

```
6010 IF position=1480 OR positio
n=1240 THEN PRINT "Brr, l'eau es
t bien trop froide": GO TO 570
6020 PRINT "Il n'y a pas d'eau i
ci !!!": GO TO 570
6499 REM
6500 REM briser
6501 REM
6510 IF b$="" THEN INPUT "Briser
 quoi ?"; LINE b$: GO TO 6510
6520 IF b$<>"verre" OR position<
>1390 THEN GO TO 750
6530 IF p(2) THEN PRINT "Je n'ai
 rien pour casser le verre !": G
O TO 570
6540 PRINT "Crac...": LET c=c-1
6550 LET p(7)=position: LET p(2)
=position
6560 FOR n=1 TO 10: BEEP 0.01,10
-n: NEXT n
6570 GO TO 570
6999 REM
7000 REM donner
7001 REM
7010 IF b$="" THEN INPUT "Donner
 quoi ?"; LINE b$: GO TO 7010
7020 IF b$<>"tresor" THEN PRINT
"Non, c'est impossible !": GO TO
 570
7030 IF position<>1450 THEN PRIN
T "Ca ne sert a rien ici!!": GO
TO 570
7040 IF p(4) THEN PRINT "Je n'ai
 pas de tresor !": GO TO 570
7050 LET c=c-1: LET monte=1180
7060 LET tresor=1: LET p(4)=1
7070 PRINT "Le guerrier accepte
de vous laisser passer."
7080 GO TO 570
7199 REM
7200 REM manger
7201 REM
7205 IF b$="" THEN INPUT "Manger
 quoi ?"; LINE b$: GO TO 7205
7207 IF b$<>"nourriture" THEN GO
 TO 750
7210 IF p(12) THEN PRINT "Je n'a
i rien a manger !!!": GO TO 570
7220 PRINT "C'est bon !!!": LET
p(12)=1480: LET c=c-1
7230 FOR n=0 TO 2
7240 POKE 23672+n,0
7250 NEXT n
7260 GO TO 570
7499 REM
7500 REM lire
7501 REM
7510 IF b$="" THEN INPUT "Lire q
uoi ?"; LINE b$: GO TO 7510
```

```
7520 IF b$<>"papier" THEN GO TO
750
7530 IF p(11) THEN PRINT "Il n'y
 a pas de papier a lire !!!": GO
 TO 570
7540 PRINT "Le papier indique qu
'une reserved'atompiles se trouv
e sur une   ile au milieu du plu
s grand lac de la planete."
7550 GO TO 570
7999 REM
8000 REM verifier
8001 REM
8010 LET trouve=0
8020 FOR n=1 TO 12
8030 IF p(n)<>0 THEN NEXT n: GO
TO 8070
8040 IF NOT trouve THEN PRINT "J
'ai ": LET trouve=1
8050 PRINT "un(e) ";o$(n)
8060 NEXT n
8070 IF NOT trouve THEN PRINT "J
e n'ai rien !!!"
8080 GO TO 570
8499 REM
8500 REM decoller
8501 REM
8510 IF p(1) OR position<>1150 T
HEN GO TO 750
8520 PRINT TAB 8; FLASH 1; INK 2
;"   VICTOIRE !!!!"
8699 REM
8700 REM fin
8701 REM
8710 FOR i=1 TO 5
8720 FOR j=20 TO 30
8730 BEEP 0.01,j
8740 NEXT j
8750 NEXT i
8760 GO TO 100
8800 REM help
8801 REM
8810 IF x>0.95 THEN PRINT "Aide
toi et le ciel t'aidera...": GO
TO 8890
8820 IF x<0.5 THEN PRINT "Allons
 courage !!!": GO TO 8890
8830 IF x=0.65 THEN PRINT "Reste
z calme et recomencez      depui
s le debut. Tout ira bien  mieux
, c'est certain !!": PAUSE 100:
RUN
8840 GO SUB regle
8890 GO TO 570
9000 BORDER 0: PAPER 0: INK 4: C
LS
9010 DIM p(12): DIM o$(12,10)
9020 DEF FN t()=INT ((65536*PEEK
 23674+256*PEEK 23673+23672)/50)
```

```
9025 LET a$=""
9030 LET c=0: LET regle=9700
9035 LET tresor=0: LET ouvert=0
9037 LET debloque=0
9040 LET position=1000
9050 LET prendre=2000: LET monte
r=2500: LET descente=2700: LET P
oser=3000
9060 LET regarder=4000: LET ecla
irer=4500: LET ouvrir=5000
9070 LET debloquer=5500: LET nag
er=6000: LET briser=6500
9080 LET donner=7000: LET manger
=7200: LET lire=7500: LET verifi
er=8000
9090 LET decoller=8500: LET fin=
8700: LET help=8800
9100 PLOT 50,0
9110 DRAW OVER 1, INK 5,0,155,87
07
9120 FOR n=5 TO 1 STEP -1
9130 PLOT 128,0
9140 DRAW OVER 1, INK 2,n*1.5,17
5-n*20,2771+PI
9150 NEXT n
9160 CIRCLE INK 4, FLASH 1,88,88
,10
9170 CIRCLE INK 4, FLASH 1,175,8
8,10
9180 FOR n=1 TO 20
9190 BEEP 0.01,10
9200 BEEP 0.01,30
9210 BEEP 0.01,20
9220 BEEP 0.01,40
9230 NEXT n
9500 FOR n=1 TO 12
9510 READ o$(n),p(n)
9520 NEXT n
9530 DATA "atompile",1480,"caill
ou",1300,"echelle",1120,"tresor"
,1,"torche",1000
9540 DATA "corde",1000,"cle",1,"
rubis",1120,"radio",1000,"anneau
",1090,"papier",1,"nourriture",1
150
9550 GO SUB regle
9560 FOR n=0 TO 2: POKE 23672+n,
0: NEXT n
9570 RETURN
9699 REM
9700 REM regle
9701 REM
9705 CLS
9710 PRINT AT 0,7;"EXPLORATION S
PATIALE"
9720 PRINT AT 1,7;"**************
*******"
```

```
9730 PRINT AT 5,5;"Alors que vou
s etiez en    mission, votre vai
sseau spatial est tombe en panne
, son atompileetant epuisee."
9740 PRINT AT 10,5;"Apres un att
errissage en   catastrophe, vous
 etes perdu surune planete etran
gere."
9750 PRINT AT 14,5;"Il vous faut
 donc trouver  au plus vite une
atompile avant de rejoindre la n
avette pour   quitter cette pla
nete maudite."
9760 PRINT AT 21,0; INK 1;"<SPAC
E> pour continuer"
9770 LET a$=INKEY$
9780 IF a$<>" " THEN GO TO 9770
9790 CLS
9800 PRINT AT 1,10;"COMMANDES"
9810 PRINT AT 2,10;"----------"
9820 PRINT : PRINT
9830 PRINT TAB 0;"n";TAB 8;"nord
";TAB 18;"s";TAB 27;"sud"
9840 PRINT TAB 0;"o";TAB 8;"oues
t";TAB 18;"e";TAB 27;"est"
9850 PRINT TAB 0;"monte";TAB 8;"
descends";TAB 18;"prends";TAB 27
;"pose"
9860 PRINT TAB 0;"revoir";TAB 8;
"regarde";TAB 18;"allume";TAB 27
;"donne"
9870 PRINT TAB 0;"ouvre";TAB 8;"
debloque";TAB 18;"nage";TAB 27;"
mange"
9880 PRINT TAB 0;"brise";TAB 8;"
lit";TAB 18;"decolle";TAB 27;"he
lp"
9890 PRINT TAB 0;"fin";TAB 8;"in
ventaire"
9900 PRINT AT 21,0; INK 1;"<SPAC
E> pour continuer"
9910 LET a$=INKEY$
9920 IF a$<>" " THEN GO TO 9910
9930 CLS
9940 RETURN
```

3.7. KAYAK

Vous allez effectuer, à bord de votre kayak ZX Spectrum, la périlleuse descente d'un torrent de montagne déchaîné. Des rochers surgissent sans cesse autour de vous et la vitesse du kayak augmente régulièrement.

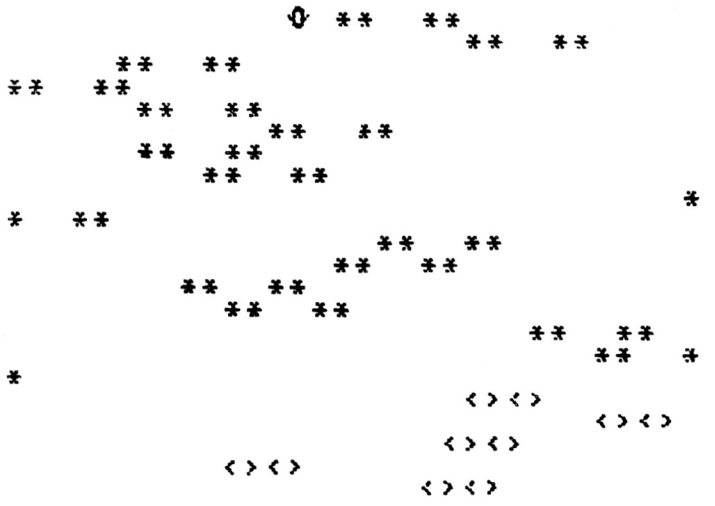
Vos réflexes seront mis à rude épreuve jusqu'à ce qu'une erreur fatale, un moment d'inattention, ne vous fasse heurter violemment un rocher, mettant ainsi fin à la descente et à la partie.

La présentation du jeu s'affiche sur l'écran. Pressez alors n'importe quelle touche pour commencer le jeu.

Vous pouvez diriger le kayak à l'aide des touches 5 et 8 (ou bien avec la manette de jeu 1).

Après la collision inévitable qui termine toute descente, pressez une touche quelconque pour voir le tableau des dix meilleurs scores. Si vous méritez de figurer dans ce tableau, vous serez poliment prié de donner votre nom.

Pressez ensuite une touche quelconque pour rejouer.



Structure du programme

- *Programme principal* : lignes 1 à 112Ø

Ligne 1Ø : appel du sous-programme de présentation.
Ligne 2Ø : appel du sous-programme d'initialisation.
Ligne 3Ø : affiche des rochers au bas de l'écran.

Ligne 4Ø : application de la technique de scrolling vue dans le chapitre 1 pour donner l'illusion que le kayak descend.

Ligne 5Ø : lecture du clavier.

Ligne 6Ø : calcul de la nouvelle position du kayak.

Ligne 7Ø : teste s'il y a collision avec un rocher.

Ligne 8Ø : affiche le kayak.

Ligne 9Ø : boucle de temporisation qui devient de plus en plus courte au fur et à mesure que le score augmente, rendant ainsi le jeu de plus en plus rapide et de plus en plus difficile.

Ligne 1ØØ : augmente le score.

Ligne 11Ø : chaque fois que le score a augmenté de 2Ø, le jeu s'accélère et la forme des rochers change.

Ligne 12Ø : retour au début de la boucle de jeu.

Ligne 1ØØØ : affichage du score. Le kayak clignote à l'endroit où il vient d'avoir un problème...

Ligne 1Ø1Ø : affichage d'un message de perte.

Ligne 1Ø2Ø à 1Ø6Ø : routine de sonorisation.

Ligne 1Ø7Ø : attente de la pression d'une touche pour continuer.

Ligne 1Ø8Ø : appel au sous-programme de traitement du tableau des dix meilleurs scores.

Ligne 1Ø9Ø : attente de la pression d'une touche pour continuer.

Ligne 11ØØ : l'écran et son bord redeviennent bleus. Effacement de l'écran.

Ligne 111Ø : appel d'une partie du sous-programme d'initialisation.

Ligne 112Ø : retour au début de la boucle de jeu.

- *Sous-programme de traitement du tableau des dix meilleurs scores* : lignes 5ØØØ à 5Ø8Ø

Ligne 5ØØØ : l'écran et son bord seront de couleur noire. Effacement de l'écran.

Ligne 5Ø1Ø : si le score figure parmi les dix meilleurs, alors appel du sous-programme de classement.

Lignes 5Ø2Ø à 5Ø7Ø : affichage du tableau d'honneur.

Ligne 5Ø8Ø : retour au programme principal.

- *Sous-programme de classement* : lignes 6ØØØ à 6Ø7Ø

Ligne 6ØØØ : sauvegarde du score.

Ligne 6Ø1Ø : demande d'entrer le nom du joueur. Seules les neuf premières lettres de ce nom seront conservées.

Lignes 6Ø2Ø à 6Ø6Ø : tri du tableau des dix meilleurs scores.

Ligne 6Ø7Ø : retour au sous-programme de traitement.

- *Sous-programme de permutation* : lignes 65ØØ à 657Ø

Lignes 65ØØ à 655Ø : permutation des éléments des tableaux p et p$.

Ligne 656Ø : indique qu'il y a eu permutation.

- *Sous-programme d'initialisation* : lignes 8ØØØ à 811Ø

Ligne 8ØØØ : effacement de l'écran.

Lignes 8Ø1Ø à 8Ø4Ø : initialisation du tableau des dix meilleurs scores et du tableau des noms.

Lignes 8ØØ à 8Ø8Ø : initialisations diverses (score, vitesse, position de départ, ...).

Ligne 81ØØ : prépare le scroll du début de partie.

Ligne 811Ø : retour au programme principal.

- *Sous-programme de présentation* : lignes 9ØØØ à 941Ø

Ligne 9ØØØ : l'écran et son bord seront de couleur noire, l'encre des caractères sera verte. Effacement de l'écran.

Ligne 9Ø1Ø : déclaration des tableaux des noms et des scores.

Ligne 9Ø2Ø : déclaration du tableau des obstacles.

Lignes 9Ø3Ø à 9Ø6Ø : initialisation du tableau des obstacles. Vous remarquerez l'utilisation des caractères de contrôle CHR$(18) et CHR$(16) qui permettent un affichage clignotant et coloré.

Lignes 9Ø7Ø à 911Ø : définition du caractère graphique " A " représentant le kayak.

Lignes 912Ø à 916Ø : apparition de trente étoiles multicolores.

Lignes 917Ø à 933Ø : affichage du nom du jeu en grosses lettres.

Ligne 934Ø : affichage d'un canot clignotant.

Lignes 935Ø à 938Ø : routine de sonorisation.

Ligne 939Ø : attente de la pression d'une touche.

Ligne 94ØØ : l'écran et son bord deviennent bleus. Effacement de l'écran.

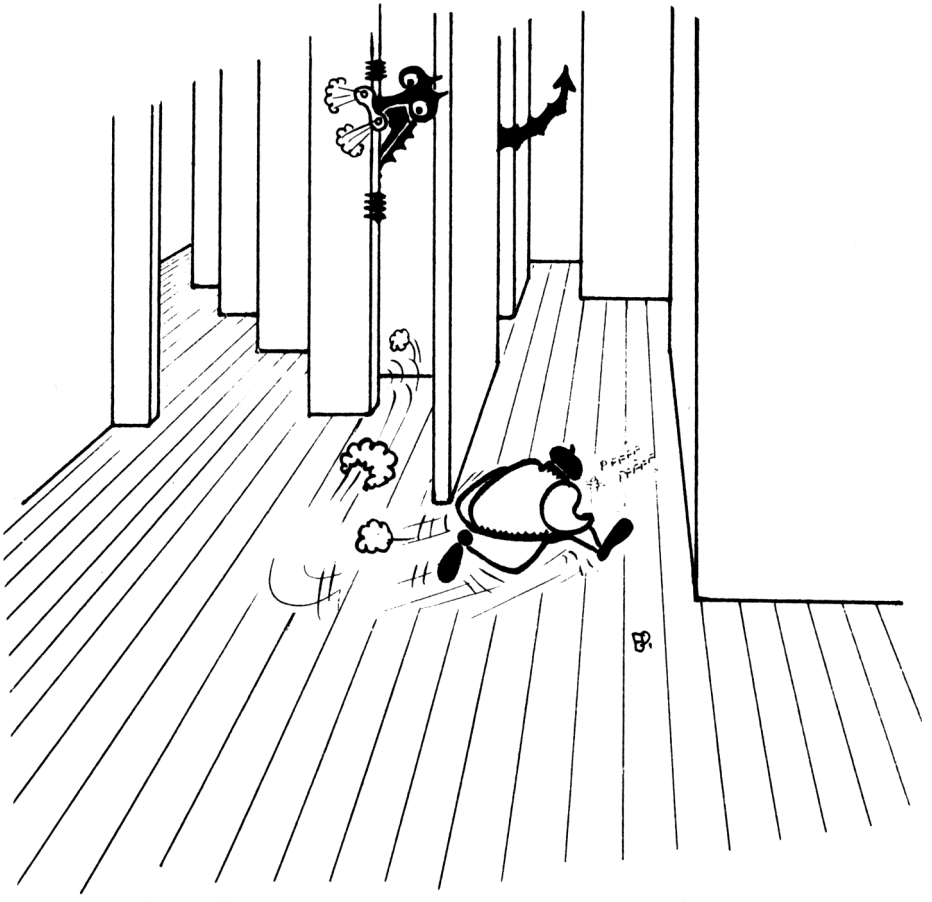
Ligne 941Ø : retour au programme principal.

```
  1 REM *********
  2 REM * KAYAK *
  3 REM *********
  4 REM
 10 GO SUB 9000
 20 GO SUB 8000
 30 PRINT AT 21,RND*31;b$
 40 POKE 23692,255: PRINT
 50 LET a$=INKEY$
 60 LET x=x+(a$="8" AND x<31)-(
a$="5" AND x>0)
 70 IF SCREEN$ (1,x)<>" " THEN
GO TO 1000
 80 PRINT AT y,x; INK 7;CHR$ (1
44)
 90 FOR j=1 TO n: NEXT j
100 LET s=s+1
110 IF s>k THEN LET k=k+20: LET
 i=INT (RND*4+1): LET b$=c$(i):
LET n=INT (n/2)
120 GO TO 30
1000 PRINT AT 1,x; INK 2; FLASH
1;CHR$ (144);AT 0,0; INK 7;"SCOR
E: ";s
1010 PRINT AT 0,20; INK 7; FLASH
 1;"CRAAAASH"
1020 FOR j=1 TO 20
1030 BEEP 0.01,10
1040 BEEP 0.01,30
1050 BEEP 0.01,20
1060 NEXT j
1070 PAUSE 4E4
1080 GO SUB 5000
1090 PAUSE 4E4
1100 BORDER 1: PAPER 1: CLS
1110 GO SUB 8050
1120 GO TO 30
5000 BORDER 0: PAPER 0: CLS
5010 IF s>p(10) THEN GO SUB 6000
5020 PRINT AT 0,10;"NOM";AT 0,2E
;"SCORE"
5030 LET q=2
5040 FOR j=1 TO 10
5050 PRINT AT q,7;p$(j);AT q,27;
p(j)
5060 LET q=q+2
5070 NEXT j
5080 RETURN
6000 LET p(11)=s
6010 INPUT "Donnez votre nom",p$
(11)
```

```
6020 LET q=0
6030 FOR j=1 TO 10
6040 IF p(j)<p(j+1) THEN GO SUB
6500
6050 NEXT j
6060 IF q<>0 THEN GO TO 6020
6070 RETURN
6500 LET u=p(j)
6510 LET u$=p$(j)
6520 LET p(j)=p(j+1)
6530 LET p$(j)=p$(j+1)
6540 LET p(j+1)=u
6550 LET p$(j+1)=u$
6560 LET q=1
6570 RETURN
8000 CLS
8010 FOR j=1 TO 11
8020 LET p(j)=0
8030 LET p$(j)="........."
8040 NEXT j
8050 LET i=0: LET s=0
8060 LET k=20: LET n=20
8070 LET x=15: LET y=0
8080 LET b$="******"
8090 CLS
8100 POKE 23692,255
8110 RETURN
9000 BORDER 0: PAPER 0: INK 4: C
LS
9010 DIM p$(11,9): DIM p(11)
9020 DIM c$(4,6)
9030 LET c$(1)="**  **"
9040 LET c$(2)=CHR$ (18)+CHR$ (1
)+CHR$ (16)+CHR$ (2)+"<><>"
9050 LET c$(3)=CHR$ (16)+CHR$ (5
)+"* *"
9060 LET c$(4)=CHR$ (18)+CHR$ (1
)+CHR$ (16)+CHR$ (3)+"<><>"
9070 FOR j=0 TO 7
9080 READ a
9090 POKE USR "A"+j,a
9100 NEXT j
9110 DATA 24,60,36,165,102,36,60
,24
9120 FOR j=1 TO 30
9140 PRINT AT RND*20,RND*31; INK
 RND*7+1;"*"
9150 BEEP 0.01,10
9160 NEXT j
9170 PRINT AT 2,2; INK 1;"K  k"
9180 PRINT AT 3,2; INK 1;"K k"
9190 PRINT AT 4,2; INK 1;"Kk"
9200 PRINT AT 5,2; INK 1;"K k";A
T 5,8; INK 2;" AA"
9210 PRINT AT 6,2; INK 1;"K  k";
AT 6,8; INK 2;"A  A"
9220 PRINT AT 7,6; INK 2;"AAAA"
9230 PRINT AT 8,8; INK 2;"A  A";
AT 8,14; INK 3;"Y  Y"
```

```
9240 PRINT AT 9,8; INK 2;"A  A";
AT 9,14; INK 3;" YY"
9250 PRINT AT 10,14; INK 3;"  Y"
9260 PRINT AT 11,14; INK 3;"  Y"
;AT 11,20; INK 4;" AA"
9270 PRINT AT 12,14; INK 3;"  Y"
;AT 12,20; INK 4;"A  A"
9280 PRINT AT 13,20; INK 4;"AAAA
"
9290 PRINT AT 14,20; INK 4;"A  A
";AT 14,26;"K  K"
9300 PRINT AT 15,20; INK 4;"A  A
";AT 15,26;"K K"
9310 PRINT AT 16,26;"KK"
9320 PRINT AT 17,26;"K K"
9330 PRINT AT 18,26;"K  K"
9340 PRINT AT 20,15; FLASH 1; IN
K 0;CHR$ (144)
9350 FOR J=1 TO 10
9360 BEEP 0.01,10+J
9370 BEEP 0.01,20+J
9380 NEXT J
9390 PAUSE 4E4
9400 BORDER 1: PAPER 1: CLS
9410 RETURN
```

3.8. LABYRINTHE



Un manque de chance incroyable vous a conduit à pénétrer dans un labyrinthe ténébreux qui s'est refermé sur vous. La seule issue maintenant possible est à l'autre bout du labyrinthe.

La nuit la plus profonde règne dans ce dédale de couloirs et vous ne pouvez avancer qu'à tâtons en vous cognant dans tous les murs.

Votre situation est d'autant plus dangereuse qu'un horrible monstre, qui demeure dans ce labyrinthe infernal, a senti votre odeur. Alléché par la perspective d'un bon festin, le monstre tente par tous les

moyens de vous rattraper pour vous dévorer. Si vous peinez trop pour sortir du labyrinthe, le monstre s'excite de plus en plus et finit par démolir tous les murs sur son passage afin de vous rattraper plus vite.

Vos malheurs ne s'arrêtent pas là. En effet, des puits très profonds que vous ne pouvez pas voir, faute de lumière, peuvent vous engloutir à tout moment !

Alors bon courage et surtout, ne perdez pas de temps !!

Dix niveaux de jeu vous sont proposés :

Niveau / *Difficulté*	1	2	3	4	5	6	7	8	9	10
labyrinthe facile	X	X	X							
labyrinthe moyen				X	X	X				
labyrinthe difficile							X	X	X	X
14 déplacements	X			X			X			
9 déplacements		X			X			X		
5 déplacements			X			X			X	X
1 puit	X	X	X	X	X	X	X	X	X	
5 puits										X

Exemple: niveau 5: labyrinthe de difficulté moyenne, neuf fois plus de déplacements que le monstre, un seul puit.

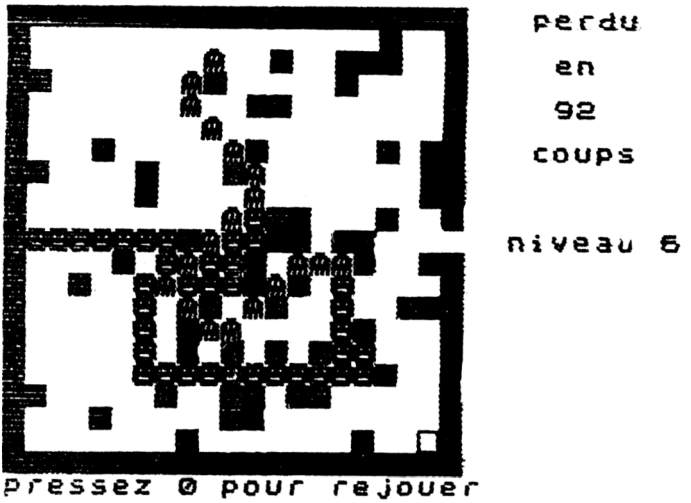
Le score est le nombre de coups mis pour sortir indemne du labyrinthe.

Chaque niveau de jeu a son propre "high score" qui garde en mémoire le meilleur score réalisé depuis le début du jeu pour le niveau concerné.

Pressez les touches 5, 6, 7 et 8 ou bien utilisez la manette de jeu 1 pour vous déplacer dans le labyrinthe.

Dès que la partie sera terminée (sortie normale, chute dans un puits ou fin tragique dans l'estomac du monstre), le labyrinthe tel qu'il était apparaîtra sous vos jeux avec les chemins que vous et le monstre avez parcourus.

Pour rejouer, presser soit Ø soit le bouton feu de la manette de jeu. La pression de toute autre touche arrête le jeu.



Structure du programme

- *Programme principal*: lignes 1 à 22Ø

Ligne 1Ø: appel du sous-programme de présentation.

Ligne 2Ø : appel du sous-programme d'initialisation.

Ligne 3Ø : appel du sous-programme de jeu.

Ligne 4Ø : teste si le joueur a gagné ou perdu.

Ligne 5Ø : cas où le joueur a gagné.

Ligne 6Ø : mise à jour éventuelle du high score pour le niveau de jeu considéré.

Ligne 8Ø : saut à la routine de sonorisation.

Ligne 9Ø : cas où le joueur est tombé dans un puit.

Ligne 1ØØ : cas où le joueur a perdu.

Lignes 11Ø à 15Ø : routine de sonorisation.

Lignes 16Ø à 185 : affichage du labyrinthe et des traces laissées par le monstre et vous-même.

Lignes 19Ø et 2ØØ : "Ø" pour rejouer.

Ligne 21Ø : arrêt du jeu si une touche autre que Ø est pressée.

Ligne 22Ø : effacement de l'écran. Retour au début du jeu.

● *Sous-programme de jeu* : lignes 5ØØØ à 523Ø

Lignes 5ØØØ à 513Ø : déplacement du joueur.

Ligne 5Ø1Ø : incrément du score.

Ligne 5Ø2Ø : au bout d'un certain temps vous ne vous déplacez que trois fois plus que le monstre (quelque soit le niveau de jeu !).

Ligne 5Ø4Ø : lecture du clavier.

Lignes 5Ø5Ø et 5Ø6Ø : calcul des nouvelles coordonnées du petit bonhomme qui vous représente.

Ligne 5Ø7Ø : cas où la sortie est atteinte.

Ligne 5Ø75 : cas où c'est vous qui avez rattrapé le monstre...

Ligne 5Ø8Ø : cas où vous êtes tombé dans un puit.

Ligne 5Ø9Ø : vous n'avez pas le droit de traverser les murs.

Ligne 51ØØ : mémorise la place prise par le petit bonhomme.

Ligne 512Ø : affichage du bonhomme à sa nouvelle position après avoir pris soin de renoircir la position d'où il vient.

Lignes 514Ø à 522Ø : déplacement du monstre.

Lignes 515Ø et 516Ø : calculent les nouvelles coordonnées du monstre en fonction des coordonnées du bonhomme.

Ligne 517Ø : dans certains cas le monstre s'énerve et ne se soucie pas d'éviter les murs.

Ligne 518Ø : cas où le monstre n'est pas encore suffisamment en colère pour tout casser sur son passage.

Ligne 519Ø : affichage du monstre à sa nouvelle position.

Ligne 52ØØ : mémorise la place prise par le monstre.

Ligne 521Ø : cas où le monstre vous a rattrapé.

Ligne 522Ø : retour à la boucle de déplacement du joueur.

Ligne 523Ø : retour au programme principal.

- *Sous-programme d'initialisation* : lignes 8ØØØ à 834Ø

Ligne 8ØØØ : initialisation de la position de départ du bonhomme.

Ligne 8Ø1Ø : initialisation de la position de départ du monstre.

Ligne 8Ø2Ø : s est la variable utilisée pour le score, r représente la complexité du labyrinthe.

Ligne 8Ø3Ø : t est le nombre de déplacements que vous faites chaque fois que le monstre avance d'une case, w est le nombre de puits.

Lignes 8Ø4Ø à 8Ø6Ø : initialisation du tableau qui contiendra les murs du labyrinthe puis les traces du monstre et du bonhomme.

Lignes 8Ø7Ø et 8Ø8Ø : demande le niveau de jeu choisi.

Ligne 8Ø9Ø : refuse tout niveau de jeu non compris entre 1 et 1Ø.

Ligne 8Ø95 : effacement de l'écran.

Ligne 81ØØ : seul le labyrinthe de niveau 1Ø a 5 puits.

Ligne 811Ø : détermine la complexité du labyrinthe en fonction du niveau de jeu.

Ligne 812Ø : détermine le nombre de déplacements autorisés en fonction du niveau de jeu.

Lignes 813Ø à 818Ø : calcul du labyrinthe en prenant soin à ce que la sortie soit dégagée (ligne 816Ø).

Lignes 819Ø à 824Ø : positionnement des puits.

Lignes 825Ø à 827Ø : limite droite du labyrinthe.

Lignes 829Ø à 832Ø: affichage du labyrinthe tout noir et de la sortie.

Ligne 833Ø: affichage du niveau de jeu choisi.

Ligne 834Ø: retour au programme principal.

- *Sous-programme de présentation*: lignes 9ØØØ à 928Ø

Ligne 9ØØØ: déclaration du tableau des high scores et du tableau destiné à contenir le labyrinthe.

Lignes 9Ø1Ø à 9Ø3Ø: initialisation des high scores.

Ligne 9Ø4Ø: initialisation de la variable b$ utilisée pour l'affichage du labyrinthe.

Lignes 9Ø5Ø à 9Ø9Ø: définition du caractère graphique "A" qui représente le petit bonhomme.

Lignes 91ØØ à 915Ø: définition du caractère graphique "B" qui représente un puit.

Ligne 916Ø à 92ØØ: définition du caractère graphique "C" qui représente le monstre.

Lignes 921Ø à 923Ø: présentation du jeu et des "adversaires".

Lignes 924Ø à 927Ø: sonorisation.

Ligne 928Ø: retour au programme principal.

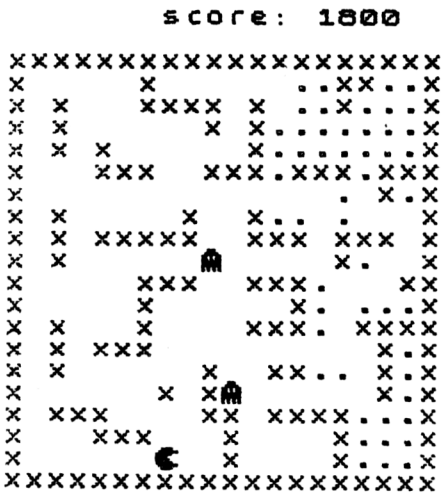
```
   1 REM **************
   2 REM * LABYRINTHE *
   3 REM **************
   4 REM
  10 GO SUB 9000
  20 GO SUB 8000
  30 GO SUB 5000
  40 IF fin<>0 THEN GO TO 90
  50 PRINT AT 0,24;"gagne";AT 2,
25;"en";AT 4,25;s;AT 6,24;"coups
"
  60 IF s<h(n) THEN LET h(n)=s
  70 PRINT AT 10,23;"niveau ";n;
AT 12,22;"high score";AT 14,25;h
(n)
  80 GO TO 110
  90 IF fin=2 THEN PRINT AT 20,2
3; FLASH 1; INK 2;"..PUIT$.."
 100 PRINT AT 0,24;"perdu";AT 2,
25;"en";AT 4,25;s;AT 6,24;"coups
"
 110 FOR i=1 TO 20
 120 BEEP 0.01,30
 130 BEEP 0.01,10
```

```
  140 BEEP 0.01,20
  150 NEXT i
  160 FOR i=1 TO 19
  170 PRINT AT i,1;p$(i)
  180 NEXT i
  185 PRINT AT 10,20;" "
  190 PRINT AT 21,0;"pressez 0 po
ur rejouer"
  200 LET a$=INKEY$: IF a$="" THE
N GO TO 200
  210 IF a$<>"0" THEN STOP
  220 CLS : GO TO 20
5000 FOR i=1 TO t
5010 LET s=s+1
5020 IF s>40+INT (RND*20) THEN L
ET t=3
5030 LET e=a: LET f=b
5040 LET a$=INKEY$
5050 LET b=b+(a$="6" AND b<19)-(
a$="7" AND b>1)
5060 LET a=a+(a$="8")-(a$="5" AN
D a>1)
5070 IF a=20 AND b=10 THEN LET f
in=0: GO TO 5230
5075 IF b=c AND a=d THEN LET fin
=1: GO TO 5230
5080 IF p$(b,a)="□" THEN LET fin
=2: GO TO 5230
5090 IF p$(b,a)="■" THEN LET a=e
: LET b=f
5100 LET p$(b,a)="■"
5120 PRINT AT f,e;"■";AT b,a; IN
K 1;"■": BEEP 0.01,10
5130 NEXT i
5140 LET k=d: LET g=c
5150 LET d=d+(d<a)-(d>a)
5160 LET c=c+(c<b)-(c>b)
5170 IF s>h(n) THEN GO TO 5200
5180 IF p$(c,d)="■" THEN LET d=k
: LET c=g
5190 PRINT AT g,k;"■";AT c,d; FL
ASH 1; INK 2;"■": BEEP 0.01,30
5200 LET p$(c,d)="■"
5210 IF c=b AND d=a THEN LET fin
=1: GO TO 5230
5220 GO TO 5000
5230 RETURN
6000 LET a=1: LET b=10
6010 LET c=1: LET d=10
6020 LET s=0: LET r=0
6030 LET t=0: LET w=1
6040 FOR i=1 TO 19
6050 LET p$(i)=""
6060 NEXT i
6070 PRINT AT 0,0;"A quel niveau
 desirez vous jouer ? (1-10)"
6080 INPUT n: LET n=INT (ABS n)
6090 IF (n-1)*(n-10)>0 THEN BEEP
 0.1,20: GO TO 6080
```

```
6095 CLS
6100 IF n=10 THEN LET w=5
6110 LET r=50*((n-4)*(n-6)<=0)+1
00*((n-7)*(n-10)<=0)
6120 LET t=5+4*(n=2 OR n=5 OR n=
8)+9*(n=1 OR n=4 OR n=7)
6130 FOR i=1 TO 50+INT (RND*r)
6140 LET e=INT (RND*19+1)
6150 LET f=INT (RND*19+1)
6160 IF (f-9)*(f-11)<0 AND (e-18
)*(e-19)=0 THEN GO TO 6140
6170 LET p$(f,e)="■": BEEP 0,01,
5
6180 NEXT i
6190 FOR i=1 TO w
6200 LET e=INT (RND*19+1)
6210 LET f=INT (RND*19+1)
6220 IF e=19 AND f=10 THEN GO TO
 6200
6230 LET p$(f,e)="□"
6240 NEXT i
6250 FOR i=1 TO 19
6260 LET p$(i,20)="■"
6270 NEXT i
6290 FOR i=0 TO 20
6300 PRINT AT i,0;b$
6310 NEXT i
6320 PRINT AT 10,20;" "
6330 PRINT AT 10,23;"niveau ";n
6340 RETURN
9000 DIM h(10): DIM p$(19,20)
9010 FOR i=1 TO 10
9020 LET h(i)=1000
9030 NEXT i
9040 LET b$="
■■"
9050 FOR i=0 TO 7
9060 READ a
9070 POKE USR "A"+i,a
9080 NEXT i
9090 DATA 126,255,219,255,255,19
5,255,126
9100 FOR i=0 TO 7
9110 POKE USR "B"+i,a
9120 READ a
9130 POKE USR "B"+i,a
9140 NEXT i
9150 DATA 255,129,129,129,129,12
9,129,255
9160 FOR i=0 TO 7
9170 READ a
9180 POKE USR "C"+i,a
9190 NEXT i
9200 DATA 56,124,214,214,254,254
,170,170
9210 PRINT AT 10,10; FLASH 1; IN
K 2;"LABYRINTHE"
9220 PRINT AT 15,5;"■";AT 15,27;
"■"
```

```
9230 PRINT AT 13,15; INK 3;"👻"
9240 FOR i=1 TO 20
9250 BEEP 0.01,i*2
9260 NEXT i
9270 CLS
9280 RETURN
```

3.9. PICMAN



Le petit PICMAN n'en croit pas ses yeux. Le labyrinthe dans lequel il vient de pénétrer est rempli de friandises délicieuses. Mais attention, PICMAN a entendu dire que d'horribles monstres habitaient dans ce labyrinthe et qu'ils pourchassaient tout PICMAN osant s'y aventurer. Si les monstres le rattrapent, c'en est fini de lui.

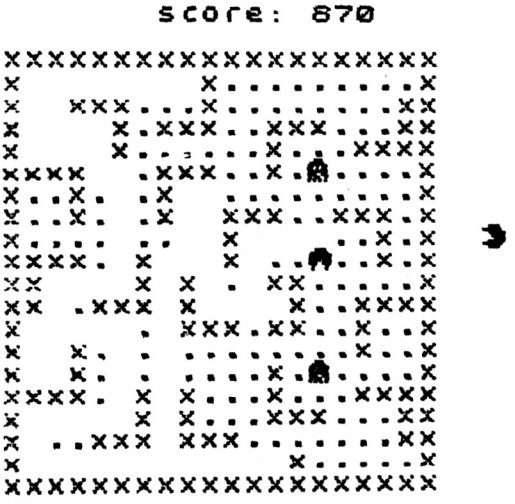
PICMAN vous fait confiance pour le guider dans le labyrinthe en évitant les monstres dangereux. Pour le diriger, vous pouvez utiliser soit les touches 5, 6, 7 et 8 soit la manette de jeu 1.

Dès que toutes les friandises d'un tableau sont mangées, un nouveau tableau apparaît. Serez-vous aussi infatigable que les monstres ?

Après la présentation du jeu, vous devez choisir le labyrinthe qui vous intéresse en donnant son numéro (1, 2 ou 3).

REMARQUE : Le programme est fait de manière à ce que vous puissiez rajouter très facilement d'autres labyrinthes ou bien modifier ceux qui existent déjà.

Le choix du labyrinthe étant fait, à vous de jouer...



Structure du programme

- *Programme principal* : lignes 1 à 4Ø9Ø

  — *bloc 1* : lignes 1 à 7Ø

Ligne 7 : appel du sous-programme de définition des caractères graphiques et de présentation du jeu.

Ligne 8 : déclaration des tableaux n et m qui contiennent les coordonnées des deux monstres, du tableau h qui contiendra les records établis dans chacun des trois labyrinthes.

Lignes 9 et 1Ø : initialisations diverses (score, numéro par "défaut" du labyrinthe, nombre de "vies").

Ligne 3Ø : demande de numéro du labyrinthe choisi. Le programme vous propose trois labyrinthes mais vous pouvez, en faisant de très petites modifications, en rajouter autant que vous le désirez.

Ligne 35 : c'est le labyrinthe 1 qui est pris par défaut.

Ligne 4Ø: seuls les numéros 1, 2 et 3 sont acceptés.

Ligne 5Ø: appel du sous-programme d'initialisation.

Ligne 6Ø: appel du sous-programme d'affichage du labyrinthe.

Ligne 7Ø: appel du sous-programme d'affichage des monstres.

— *bloc 2*: boucle de jeu du joueur, lignes 1ØØØ à 11ØØ

Lignes 1Ø2Ø à 1Ø5Ø: déplacement du petit glouton. Celui-ci ne peut pas, bien entendu, traverser les murs.

Ligne 1Ø6Ø: teste si une friandise vient d'être mangée et augmente le score si c'est le cas.

Ligne 1Ø62: teste s'il reste encore des friandises. Si ce n'est pas le cas (q = Ø) alors un nouveau tableau est affiché (ligne 1Ø65).

Lignes 1Ø7Ø et 1Ø8Ø: testent si les monstres ont réussi à rattraper le glouton.

Ligne 1Ø9Ø: affichage de PICMAN.

Ligne 111Ø: teste si c'est encore à PICMAN de se déplacer ou bien si c'est au tour des monstres de bouger. En pratique le glouton bouge environ deux fois plus que les monstres et va donc deux fois plus vite.

Vous remarquerez que cette boucle de jeu contient beaucoup de tests. Pour permettre au jeu d'être suffisamment rapide, bien qu'écrit en Basic, il a donc été nécessaire de mettre les tests en cascade (IF ... THEN IF ... THEN).

— *bloc 3*: déplacements des monstres, lignes 2ØØØ à 262Ø

Ligne 2Ø1Ø: choix du monstre qui va se déplacer.

Ligne 2Ø3Ø: affichage du monstre en utilisant la technique du PRINT OVER (voir chapitre 1).

Lignes 2Ø5Ø et 2Ø6Ø: déplacement horizontal du monstre en fonction de la position de PICMAN.

Lignes 2Ø7Ø et 2Ø8Ø: déplacement vertical du monstre en fonction de la position de PICMAN.

De même que PICMAN, les monstres n'ont pas le droit de traverser les murs. Par contre ils peuvent se déplacer en diagonale.

Ligne 2Ø9Ø : teste si PICMAN est rattrapé.

Ligne 211Ø : affichage du monstre (cf. ligne 2Ø3Ø).

Ligne 212Ø : retour à la boucle de jeu du joueur.

Lignes 252Ø à 262Ø : même chose que les lignes 2Ø2Ø à 212Ø mais pour le monstre 2. La solution d'écrire deux fois la même chose à quelques différences près, plutôt que de paramétrer le déplacement des monstres, a été choisie pour obtenir une rapidité maximum d'exécution.

— *bloc 4* : perte, lignes 3ØØØ à 4Ø9Ø

Ligne 3Ø1Ø : le joueur perd une "vie".

Ligne 3Ø2Ø : la partie est terminée dès que trois PICMANs ont été mangés.

Ligne 3Ø6Ø : enlève un PICMAN au bord du labyrinthe.

Ligne 3Ø8Ø : appel du sous-programme d'initialisation.

Ligne 3Ø9Ø : retour au début de la boucle de jeu du joueur.

Ligne 391Ø : met à jour, s'il y a lieu de le faire, le record réalisé pour le labyrinthe considéré.

Ligne 392Ø : affichage du record.

Lignes 4ØØØ à 4Ø5Ø : routine de sonorisation.

Ligne 4Ø6Ø : attente de la pression d'une touche pour continuer.

Ligne 4Ø7Ø : effacement de l'écran. Demande de presser ENTER pour rejouer.

Ligne 4Ø8Ø : le jeu s'arrête si le joueur entre une chaîne non vide.

Ligne 4Ø9Ø : début d'une nouvelle partie (les records sont conservés).

- *Sous-programme de décomposition de PICMAN* : lignes 45ØØ à 458Ø

PICMAN, qui vient de se faire rattraper par un monstre, disparaît en trois temps.

- *Sous-programme de sonorisation* : lignes 46ØØ à 467Ø
- *Sous-programme de définition de caractères graphiques et de présentation du jeu* : lignes 5ØØØ à 634Ø

Lignes 5Ø1Ø à 52ØØ : définition de PICMAN dans ses différentes positions (caractères graphiques "A", "B", "C" et "D").

Lignes 6ØØØ à 6Ø5Ø : définition des monstres (caractère graphique "M").

Lignes 6Ø6Ø à 615Ø : définition des caractères représentant PICMAN en train de disparaître (caractères graphiques "E" et "F").

Ligne 616Ø : l'écran et son bord seront de couleur noir ; l'encre des caractères sera verte. Effacement de l'écran.

Lignes 617Ø à 623Ø : affichage en grosses lettres multicolores du nom du jeu.

Lignes 624Ø à 629Ø : affichage de deux monstres qui se précipitent sur un pauvre petit PICMAN affolé.

Ligne 63ØØ : les monstres "mangent" PICMAN.

Ligne 632Ø : les monstres sont fiers de leur victoire facile et en clignotent de joie.

Ligne 634Ø : retour au programme principal.

- *Sous-programme d'affichage d'un labyrinthe* : lignes 7ØØØ à 768Ø

Ligne 7ØØ5 : effacement de l'écran.

Ligne 7Ø1Ø : affichage au bord du terrain des PICMANs représentant les "vies" du joueur.

Ligne 7Ø2Ø : initialisation de la variable a$ utilisée pour l'affichage d'un labyrinthe.

Ligne 7Ø3Ø : affichage de la première ligne et de la dernière ligne d'un labyrinthe.

Lignes 7Ø4Ø à 7Ø45 : saut à l'affichage du labyrinthe choisi. Vous remarquerez qu'il est très facile d'ajouter de nouveaux labyrinthes. La variable q représente le nombre de friandises que contient le labyrinthe.

Lignes 7Ø47 à 7225 : affichage du labyrinthe 1.

Lignes 73ØØ à 748Ø : affichage du labyrinthe 2.

Lignes 75ØØ à 768Ø : affichage du labyrinthe 3.

La solution d'afficher le labyrinthe de cette façon a été retenue car elle est très rapide mais il est vrai qu'elle est peu élégante.

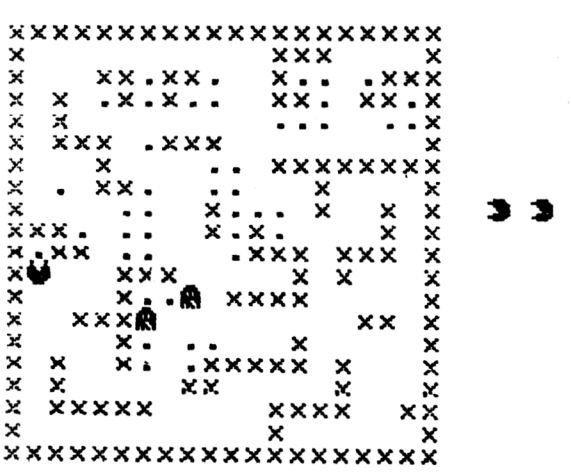
- *Sous-programme d'initialisation* : lignes 8000 à 8050

Ligne 8005 : position de départ de PICMAN.

Ligne 8010 : positions de départ des monstres.

Ligne 8050 : retour au programme principal.

- *Sous-programme d'affichage des monstres* : lignes 9000 à 9020




```
    1 REM ***********
    2 REM * PICMAN *
    3 REM ***********
    4 REM
    7 GO SUB 5000
    8 DIM n(4): DIM m(4): DIM h(3
)
    9 LET s=0
   10 LET r=1: LET t=3
   20 LET p$="B"
   30 INPUT "quel labyrinthe desi
rez vous ?",r$
   35 IF r$="" THEN LET r$="1"
   37 LET r=VAL r$
   40 IF (r-1)*(r-2)*(r-3)<>0 THE
N PRINT "impossible": GO TO 30
   50 GO SUB 8000
   60 GO SUB 7000
   70 GO SUB 9000
1000 REM deplacement du joueur
1010 PRINT AT pv,ph;" "
1020 IF INKEY$="à" THEN IF SCREE
N$ (pv,ph+1)<>"x" THEN LET ph=ph
+1: LET p$="A": GO TO 1060
```

```
1030 IF INKEY$="5" THEN IF SCREE
N$ (pv,ph-1)<>"x" THEN LET ph=ph
-1: LET p$="B": GO TO 1060
1040 IF INKEY$="6" THEN IF SCREE
N$ (pv+1,ph)<>"x" THEN LET pv=pv
+1: LET p$="C": GO TO 1060
1050 IF INKEY$="7" THEN IF SCREE
N$ (pv-1,ph)<>"x" THEN LET pv=pv
-1: LET p$="D"
1060 IF SCREEN$ (pv,ph)="." THEN
 LET s=s+10: LET q=q-1
1062 IF q>0 THEN GO TO 1070
1065 GO TO 50
1070 IF ph=n(1) THEN IF pv=n(2)
THEN GO SUB 4500: PRINT AT n(4),
n(3); OVER 1;"M": GO TO 3000
1080 IF ph=n(3) THEN IF pv=n(4)
THEN GO SUB 4500: PRINT AT n(2),
n(1); OVER 1;"M": GO TO 3000
1090 PRINT AT pv,ph;; INK 1; BRI
GHT 1;p$: BEEP 0.005,10
1100 IF RND>0.5 THEN GO TO 1000
1110 PRINT AT 0,10; INK 7;"score
: ";s
2000 REM deplacement monstre 1
2010 IF INT (RND*2+1)=2 THEN GO
TO 2500
2020 LET m(1)=n(1): LET m(2)=n(2
)
2030 PRINT AT m(2),m(1); OVER 1;
"M"
2050 IF ph<n(1) THEN IF SCREEN$
(n(2),n(1)-1)<>"x" THEN LET n(1)
=n(1)-1: GO TO 2070
2060 IF ph>n(1) THEN IF SCREEN$
(n(2),n(1)+1)<>"x" THEN LET n(1)
=n(1)+1
2070 IF pv<n(2) THEN IF SCREEN$
(n(2)-1,n(1))<>"x" THEN LET n(2)
=n(2)-1: GO TO 2085
2080 IF pv>n(2) THEN IF SCREEN$
(n(2)+1,n(1))<>"x" THEN LET n(2)
=n(2)+1
2090 IF ph=n(1) THEN IF pv=n(2)
THEN GO SUB 4500: PRINT AT n(4),
n(3); OVER 1;"M": GO TO 3000
2110 PRINT AT n(2),n(1); INK 2;
BRIGHT 1; OVER 1;"M": BEEP 0.005
,30
2120 GO TO 1000
2520 LET m(3)=n(3): LET m(4)=n(4
)
2530 PRINT AT m(4),m(3); OVER 1;
"M"
2550 IF ph<n(3) THEN IF SCREEN$
(n(4),n(3)-1)<>"x" THEN LET n(3)
=n(3)-1: GO TO 2570
2560 IF ph>n(3) THEN IF SCREEN$
(n(4),n(3)+1)<>"x" THEN LET n(3)
=n(3)+1
```

```
2570 IF pv<n(4) THEN IF SCREEN$
(n(4)-1,n(3))<>"x" THEN LET n(4)
=n(4)-1: GO TO 2585
2580 IF pv>n(4) THEN IF SCREEN$
(n(4)+1,n(3))<>"x" THEN LET n(4)
=n(4)+1
2590 IF ph=n(3) THEN IF pv=n(4)
THEN GO SUB 4500: PRINT AT n(2),
n(1); OVER 1;"M": GO TO 3000
2610 PRINT AT n(4),n(3); INK 2;
BRIGHT 1; OVER 1;"M": BEEP 0.005
,60
2620 GO TO 1000
3000 REM perte
3010 LET t=t-1
3020 IF t=0 THEN GO TO 3900
3060 LET p$="B": PRINT AT 10,25;
" ";AT 10,25;: FOR i=1 TO t
-1: PRINT INK 1;p$;" ";: NEXT i
3080 GO SUB 8000
3090 GO TO 70
3910 IF s>h(r) THEN LET h(r)=s
3920 PRINT AT 10,25; INK 1;"RECO
RD:";AT 12,25; FLASH 1;h(r)
4000 FOR j=1 TO 5
4010 FOR i=20 TO 10 STEP -2
4020 BEEP 0.01,i
4030 NEXT i
4040 BEEP 0.4,20
4050 NEXT j
4060 PAUSE 4E4: PAUSE 4E4
4070 CLS : INPUT "pressez ENTER
pour rejouer",r$
4080 IF r$<>"" THEN STOP
4090 GO TO 0
4500 PRINT AT pv,ph; INK 1; BRIG
HT 1;"E"
4510 GO SUB 4600
4520 PRINT AT pv,ph; INK 1; BRIG
HT 1;"F"
4530 GO SUB 4600
4570 PRINT AT pv,ph;" "
4580 RETURN
4610 FOR i=1 TO 3
4620 FOR j=20 TO 10 STEP -1
4630 BEEP 0.005,j
4640 NEXT j
4650 NEXT i
4670 RETURN
5000 REM definition de PICMAN
5010 FOR i=0 TO 7
5020 READ a
5030 POKE USR "A"+i,a
5040 NEXT i
5050 DATA 60,127,252,240,248,252
,127,60
5060 FOR i=0 TO 7
5070 READ a
5080 POKE USR "B"+i,a
5090 NEXT i
```

```
5100 DATA 60,254,63,31,15,63,254
,60
5110 FOR i=0 TO 7
5120 READ a
5130 POKE USR "C"+i,a
5140 NEXT i
5150 DATA 60,126,255,255,247,231
,66,6F
5160 FOR i=0 TO 7
5170 READ a
5180 POKE USR "D"+i,a
5190 NEXT i
5200 DATA 66,66,231,247,255,255,
126,60
6000 REM definition des monstres
6010 FOR i=0 TO 7
6020 READ a
6030 POKE USR "H"+i,a
6040 NEXT i
6050 DATA 56,124,214,214,254,254
,170,170
6060 FOR i=0 TO 7
6070 READ a
6080 POKE USR "E"+i,a
6090 NEXT i
6100 DATA 40,136,145,40,28,52,16
+,16+
6110 FOR i=0 TO 7
6120 READ a
6130 POKE USR "F"+i,a
6140 NEXT i
6150 DATA 0,0,0,0,0,60,126,255
6160 BORDER 0: PAPER 0: INK 4: C
LS
6170 PRINT AT 3,0; INK 4;"
"; INK 1;"
"; INK 5;"
"; IN
K 2;"
"; INK 3;"
"; IN
K 6;"
"
6180 PRINT AT 4,0; INK 4;"
"; INK 1;"
"; INK 5;"
"; IN
K 2;"
"; INK 3;"
"; IN
K 6;"
"
6190 PRINT AT 5,0; INK 4;"
"; INK 1;"
"; INK 5;"
"; IN
K 2;"
"; INK 3;"
"; IN
K 6;"
"
6200 PRINT AT 6,0; INK 4;"
"; INK 1;"
"; INK 5;"
"; IN
K 6;"
"
6210 PRINT AT 7,0; INK 4;"
"; INK 1;"
"; INK 5;"
"; IN
K 6;"
"
6220 PRINT AT 8,0; INK 4;"
"; INK 1;"
"; INK 5;"
"; IN
K 6;"
"
```

```
6230 PRINT AT 9,0; INK 4;"█    "
; INK 1;"█████"; INK 5;"█████";
INK 2;"█   █"; INK 3;"█   █"; IN
K 6;"█   █"
6240 FOR i=0 TO 15
6250 PRINT AT 17,i; INK 2; BRIGH
T 1;"M": BEEP 0.01,30: PRINT AT
17,i;" "
6260 PRINT AT 17,15; INK 1; BRIG
HT 1;"A": BEEP 0.01,10
6270 PRINT AT 17,30-i; INK 2; BR
IGHT 1;"M": BEEP 0.01,50: PRINT
AT 17,30-i;" "
6280 PRINT AT 17,15; INK 1: BRIG
HT 1;"B": BEEP 0.01,10
6290 NEXT i
6300 LET py=17: LET ph=15: GO SU
B 4600
6320 PRINT AT 17,15; INK 2; BRIG
HT 1; FLASH 1;"M"
6340 RETURN
7000 REM labyrinthe
7005 CLS
7010 LET p$="D": PRINT AT 10,25;
"      ";AT 10,25;: FOR i=1 TO t
-1: PRINT INK 1;p$;" ";: NEXT i
7020 LET a$="xxxxxxxxxxxxxxxxxxx
x"
7030 PRINT AT 2,3;a$;AT 21,3;a$
7040 IF r=2 THEN LET q=232: GO T
O 7300
7045 IF r=3 THEN LET q=232: GO T
O 7500
7047 LET q=232
7050 PRINT AT 3,3;"x...........x
xx....x"
7060 PRINT AT 4,3;"x...xx.xx...x
....xxx"
7070 PRINT AT 5,3;"x.x..x.x....x
x..xx.x"
7080 PRINT AT 6,3;"x.x..........
......x"
7090 PRINT AT 7,3;"x.xxx..xxx...
......x"
7100 PRINT AT 8,3;"x...x.......x
xxxxxxx"
7110 PRINT AT 9,3;"x...xx.......
.x....x"
7120 PRINT AT 10,3;"x........x..
..x..x.x"
7130 PRINT AT 11,3;"xxx......x.x
.....x.x"
7140 PRINT AT 12,3;"x.xx.......x
xx.xxx.x"
7150 PRINT AT 13,3;"x....xxx....
.x.x...x"
7160 PRINT AT 14,3;"x....x....xx
xx.....x"
```

```
7170 PRINT AT 15,3;"x..xxx.....
....xx.x"
7180 PRINT AT 16,3;"x....x.....
.x.....x"
7190 PRINT AT 17,3;"x.x..x...xxx
xx.x...x"
7200 PRINT AT 18,3;"x.x.....xx..
...x...x"
7210 PRINT AT 19,3;"x.xxxxx.....
xxxx..xx"
7220 PRINT AT 20,3;"x..........
x.....x"
7226 RETURN
7300 PRINT AT 3,3;"x.....x.....
..xx..x"
7310 PRINT AT 4,3;"x.x...xxxx.x.
..x...x"
7320 PRINT AT 5,3;"x.x......x.x.
.....x"
7330 PRINT AT 6,3;"x.x.x.....x.
.....x"
7340 PRINT AT 7,3;"x...xxx..xxx.
xxx.xxx"
7350 PRINT AT 8,3;"x..........
....x.x"
7360 PRINT AT 9,3;"x.x.....x..x.
.....x"
7370 PRINT AT 10,3;"x.x.xxxxx..x
xx.xxx.x"
7380 PRINT AT 11,3;"x.x........
...x...x"
7390 PRINT AT 12,3;"x.....xxx..x
xx....xx"
7400 PRINT AT 13,3;"x.....x....
.x.....x"
7410 PRINT AT 14,3;"x.x...x....x
xx..xxxx"
7420 PRINT AT 15,3;"x.x.xxx....
.....x.x"
7430 PRINT AT 16,3;"x.x.....x..
xx....x.x"
7440 PRINT AT 17,3;"x.....x.x..
.....x.x"
7450 PRINT AT 18,3;"x.xxx....xx.
xxxx...x"
7460 PRINT AT 19,3;"x...xxx...x.
...x...x"
7470 PRINT AT 20,3;"x........x.
...x...x"
7480 RETURN
```

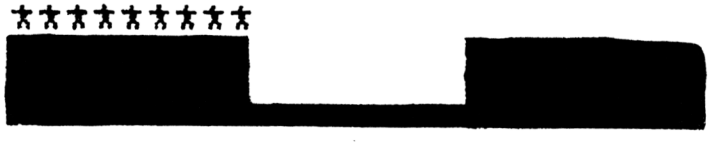
3.10. TARZAN

Neuf explorateurs courageux doivent traverser un grand précipice au fond duquel coule un torrent déchaîné.

Le seul moyen de rejoindre l'autre rive est d'attraper au bon moment une des lianes qui se balancent dans le vent, mais quand on ne s'appelle pas Tarzan, ce n'est pas facile. Gare aux chutes fatales !

Combien ferez-vous traverser d'explorateurs indemnes ?

Pour tenter d'attraper une liane, il suffit de presser une touche quelconque.



Structure du programme

- *Programme principal* : lignes 1 à 7Ø

Ligne 1Ø : appel du sous-programme de présentation.
Ligne 2Ø : appel du sous-programme d'initialisation.
Ligne 3Ø : appel du sous-programme de jeu.

Ligne 45 : affichage des explorateurs tombés au fond du précipice (s'il y en a...).

Lignes 5Ø et 6Ø : fin de partie classique.

- *Sous-programme d'"initialisation" d'un explorateur* : lignes 1ØØØ à 1Ø7Ø

Ce sous-programme met en "lice" un nouvel explorateur chaque fois que le précédent est soit arrivé à bon port, soit tombé au fond du précipice.

Ligne 1ØØØ : coordonnées de départ de l'explorateur.

Ligne 1Ø1Ø : coordonnées de dessin.

Ligne 1Ø2Ø : l'explorateur n'est pas en mouvement (tarzan = Ø) et la liane est "vide" (charge = Ø).

Ligne 1Ø4Ø : enlève un explorateur de la file de départ.

Ligne 1Ø5Ø : arrêt du jeu quand il n'y a plus d'explorateurs. C'est le seul point de sortie de la boucle de jeu principale.

Ligne 1Ø6Ø : dessin de l'explorateur.

- *Sous-programme de chute* : lignes 2ØØØ à 21ØØ

Lignes 2ØØØ à 2Ø7Ø : chute de l'explorateur avec les sons appropriés.

Ligne 2Ø95 : appel au sous-programme d'"initialisation" d'un explorateur.

- *Sous-programme d'affichage d'un explorateur* : lignes 3ØØØ à 3Ø9Ø

Ce sous-programme est particulièrement intéressant dans la mesure où il permet d'afficher un explorateur en n'importe quel emplacement de l'écran. L'utilisation d'un caractère redéfini, tel que celui qui est employé pour la file de départ, n'aurait pas permis de suivre correctement le déplacement de la liane.

- *Sous-programme de déplacement d'un explorateur* : lignes 4ØØØ à 4Ø7Ø

Ligne 4ØØØ : mise à 1 de tarzan pour indiquer que l'explorateur se déplace.

Lignes 4Ø1Ø et 4Ø2Ø : affichage de l'explorateur.

Lignes 4Ø3Ø et 4Ø4Ø : déplacement et affichage de l'explorateur.

Ligne 4Ø5Ø : teste si l'explorateur a réussi à attraper la liane (distance faible par rapport à l'extrémité de la liane et bon sens de déplacement de celle-ci). Si c'est le cas, l'indicateur charge est mis à 1 pour signaler que désormais un explorateur est accroché au bout de la liane.

Ligne 4Ø6Ø : teste si l'explorateur tombe au fond du précipice.

- *Sous-programme de "dispatching"* : lignes 5ØØØ à 5Ø2Ø

Ligne 5Ø1Ø : cas où la liane est occupée.

Ligne 5Ø2Ø : cas où le joueur presse une touche ou un explorateur est en train de se déplacer.

- *Sous-programme de déplacement de la liane* : lignes 6ØØØ à 6Ø7Ø

Ligne 6ØØØ : drapeau sert à indiquer si la liane peut être attrapée par l'explorateur.

Ligne 6Ø1Ø : la boucle est paramétrée pour permettre d'utiliser le même sous-programme quel que soit le sens de déplacement de la liane.

Ligne 6Ø2Ø : affichage de la liane. Les coordonnées de l'extrémité de la liane sont placées dans un tableau, initialisé au cours de la présentation du jeu, afin d'accélérer le mouvement de la liane.

Ligne 6Ø3Ø : appel du sous-programme de dispatching.

Ligne 6Ø4Ø : effacement de la liane.

Ligne 6Ø5Ø : cas où un explorateur est en déplacement.

- *Sous-programme de jeu* : lignes 7ØØØ à 7Ø9Ø

Ligne 7ØØØ : la liane va se déplacer de droite à gauche.

Ligne 7Ø1Ø : appel du sous-programme de déplacement de la liane.

Ligne 7Ø2Ø : si un explorateur est suspendu à la liane, on va le déposer sain et sauf sur la rive.

Ligne 7Ø3Ø : la ligne va se déplacer de gauche à droite.

Ligne 7Ø4Ø : appel du sous-programme de déplacement de la liane.

Ligne 7Ø5Ø : retour au début de la boucle de jeu. Comme cela a été dit plus haut, la sortie du jeu ne peut se faire que s'il n'y a plus d'explorateurs lors de l'appel du sous-programme 1ØØØ.

Ligne 7Ø6Ø : cas où l'explorateur est arrivé sur la rive.

Lignes 7Ø7Ø et 7Ø8Ø : affichage de l'explorateur dans sa position d'arrivée.

Ligne 7Ø87 : appel du sous-programme "d'initialisation" d'un explorateur.

Ligne 7Ø88 : retour au milieu de la boucle de jeu pour continuer le mouvement de la liane.

Ligne 7Ø9Ø : ce RETURN ne sert à rien. Il est là par "purisme".

- *Sous-programme d'initialisation* : lignes 8ØØØ à 812Ø

Ligne 8ØØØ : il y a neuf explorateurs au départ et pour l'instant aucun à l'arrivée (sauve = Ø).

Lignes 8Ø1Ø à 8Ø5Ø : redéfinition du caractère graphique A.

Lignes 8Ø7Ø à 8Ø9Ø : affichage de la file de départ.

Ligne 811Ø : appel du sous-programme d'"initialisation" d'un explorateur.

- *Sous-programme de présentation* : lignes 9ØØØ à 929Ø

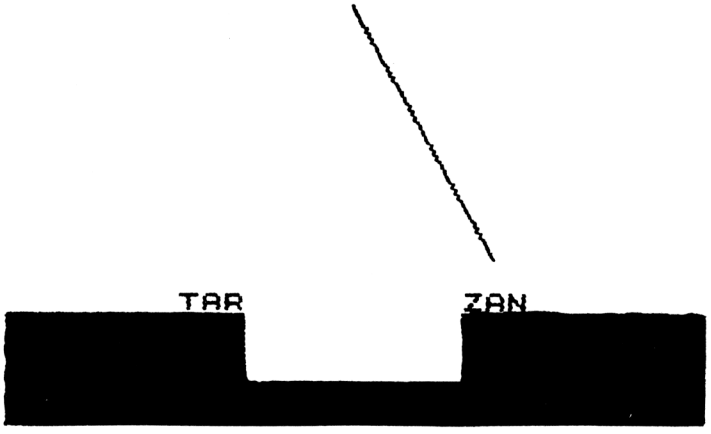
Ligne 9ØØØ : l'écran et son bord sont blancs ; l'encre des caractères est noire.

Ligne 9Ø1Ø : dimensionnement des tableaux x et y qui vont contenir les coordonnées des points situés sur la trajectoire de l'extrémité de la liane. Ces tableaux sont initialisés par les lignes 9Ø2Ø à 9Ø7Ø. Aucun calcul n'étant plus nécessaire, le mouvement de la liane sera assez rapide.

Lignes 9Ø8Ø et 91ØØ : initialisation de variables utilisées pour la présentation du jeu.

Ligne 911Ø : appel du sous-programme d'affichage du précipice.

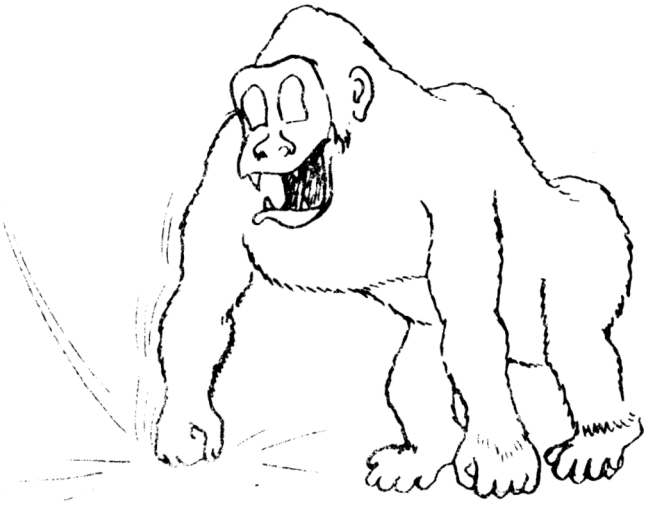
Lignes 913Ø à 92ØØ : présentation du jeu en faisant traverser le précipice par le mot TARZAN.



Lignes 923Ø à 927Ø: routine de sonorisation.

- *Sous-programme d'affichage du précipice*: lignes 95ØØ à 957Ø
- *Sous-programme de déplacement de la chaîne pour la présentation*: lignes 99ØØ à 995Ø.

```
    1 REM **********
    2 REM * TARZAN *
    3 REM **********
    4 REM
   10 GO SUB 9000
   20 GO SUB 6000
   30 GO SUB 7000
   40 PLOT OVER 1;127,150: PRINT
AT 19,11;
   45 FOR i=1 TO mort: PRINT INK
2; FLASH 1;"†";: NEXT i
   50 INPUT "Voulez-vous rejouer
?";r$
   60 IF r$="" OR r$(1)="o" THEN
RUN
   70 STOP
1000 LET ty=50: LET tx=240
1010 LET dx=tx: LET dy=ty
1020 LET tarzan=0: LET charge=0
1040 LET q=q+1: PRINT AT q,31;"
"
1050 IF homme=0 THEN GO TO 40
1060 GO SUB 3010
1070 RETURN
2000 GO SUB 3010
2010 LET ty=ty-2
2020 LET dy=ty
2030 GO SUB 3010
```



```
2040 BEEP 0.01,ty/2
2050 GO SUB 3010
2060 IF ty>30 THEN GO TO 2010
2070 BEEP 1,-10
2085 LET homme=homme-1
2090 LET mort=mort+1
2095 GO SUB 1000
2100 RETURN
3000 LET dx=127+x(i): LET dy=150
+y(i)-2
3010 PLOT OVER 1;dx,dy
3020 PLOT OVER 1;dx,dy-1
3030 PLOT OVER 1;dx-3,dy-2
3040 DRAW OVER 1;6,0
3050 PLOT OVER 1;dx,dy-3
3060 DRAW OVER 1;2,-4
3070 PLOT OVER 1;dx+1,dy-3
3080 DRAW OVER 1;-2,-4
3090 RETURN
4000 LET tarzan=1
4010 LET dx=tx: LET dy=ty
4020 GO SUB 3010
4030 LET tx=tx-10: LET dx=tx
4040 GO SUB 3010
4050 IF ABS (tx-127-x(i))<5 AND
drapeau=1 THEN LET charge=1: GO
SUB 3010: GO TO 4070
4060 IF tx<160 THEN GO SUB 2000
4070 RETURN
5000 IF charge=1 THEN GO SUB 300
0: GO TO 5020
5010 IF INKEY$<>"" OR tarzan=1 T
HEN GO SUB 4000
```

```
5020 RETURN
6000 LET drapeau=0
6010 FOR i=a TO b STEP c
6020 PLOT 127,150: DRAW x(i),y(i
)
6030 GO SUB 5000
6040 PLOT 127,150: DRAW OVER 1;x
(i),y(i)
6050 IF tarzan=1 THEN LET drapea
u=(c=1): GO SUB 5000
6060 NEXT i
6070 RETURN
7000 LET a=1: LET b=11: LET c=1
7010 GO SUB 6000
7020 IF charge=1 THEN GO TO 7060
7030 LET a=11: LET b=1: LET c=-1
7040 GO SUB 6000
7050 GO TO 7000
7060 LET sauve=sauve+1
7070 LET dx=(sauve-1)*10+5: LET
dy=50
7080 GO SUB 3010
7085 LET homme=homme-1
7087 GO SUB 1000
7088 GO TO 7030
7090 RETURN
8000 LET homme=9: LET sauve=0
8010 FOR i=0 TO 7
8020 READ a
8030 POKE USR "A"+i,a
8040 NEXT i
8050 DATA 16,16,254,24,24,24,36,
36
8070 FOR q=6 TO 14
8080 PRINT AT q,31;"▲"
8090 NEXT q
8100 LET q=5: LET mort=0
8110 GO SUB 1000
8120 RETURN
9000 BORDER 7: PAPER 7: INK 0: C
LS
9010 DIM x(11): DIM y(11)
9020 LET i=0
9030 FOR j=-PI/6 TO PI/6 STEP 0.
1
9040 LET i=i+1
9050 LET x(i)=-105*SIN j
9060 LET y(i)=-105*COS j
9070 NEXT j
9080 LET a$="[redacted]"
9090 LET b$="TARZAN": LET c$=""
9100 LET d=11
9110 GO SUB 9500
9120 PRINT AT 16,21;b$
9130 LET a=1: LET b=11: LET c=1
9140 BEEP 0.01,10: GO SUB 9900
9150 LET c$=c$+b$(1): LET b$=b$(
2 TO )
9160 LET d=d-1
```

```
9170 PRINT AT 16,21;"          ";AT
16,21;b$;AT 16,d;c$
9180 LET a=11: LET b=1: LET c=-1
9190 BEEP 0.01,20: GO SUB 9900
9200 IF b$<>"" THEN GO TO 9130
9210 PRINT AT 16,d; INK 2; FLASH
 1;c$
9220 PLOT OVER 1;127,150
9230 FOR i=1 TO 10
9240 FOR j=20 TO 30
9260 BEEP 0.01,j
9260 NEXT j
9270 NEXT i
9280 PRINT AT 16,d;"
9290 RETURN
9500 PRINT AT 17,0;
9510 FOR i=17 TO 21
9520 PRINT TAB 0; INK 4;a$;TAB 2
1; INK 4;a$
9530 NEXT i
9640 LET a$=a$(2 TO )
9550 PRINT AT 20,11; INK 1;a$
9560 PRINT AT 21,11; INK 1;a$
9570 RETURN
9900 FOR i=a TO b STEP c
9910 PLOT 127,150: DRAW x(i),y(i
)
9930 PLOT 127,150: DRAW OVER 1;x
(i),y(i)
9940 NEXT i
9950 RETURN
```

4

Adaptation des jeux pour ZX 81

Dernier-né de la société SINCLAIR, le ZX Spectrum peut être considéré comme le grand frère du ZX 81. Bien que doté de possibilités beaucoup plus grandes en particulier au niveau des couleurs, des sons et de la capacité mémoire, le ZX Spectrum reste très proche du ZX 81.

Le Basic du ZX 81 étant un sous-ensemble du Basic du ZX Spectrum, il est donc possible d'adapter très facilement des programmes écrits pour ZX 81 au ZX Spectrum.

Ce chapitre a essentiellement pour but de vous montrer comment transformer simplement les trente cinq jeux du livre "ZX 81 à la conquête des jeux" (du même auteur aux Éditions Eyrolles) pour les faire "tourner" sur ZX Spectrum. Cela vous permettra ainsi de disposer d'une bibliothèque de jeux très variés : combat de chars, squash, B 52, Cube, ...

Après un bref aperçu des principales différences entre ZX 81 et ZX Spectrum, vous trouverez un exemple complet d'adaptation du jeu FIL D'ARIANE.

4.1. DIFFÉRENCES ENTRE ZX 81 ET ZX SPECTRUM

• *Fast-Slow* :

Le ZX 81 a deux modes de fonctionnement : FAST (traitement rapide sans affichage) et SLOW (traitement lent avec affichage). Le ZX Spectrum n'a pas les instructions FAST et SLOW de son prédécesseur puisqu'il fonctionne en permanence dans un mode équivalent au mode FAST en rapidité, tout en gérant l'affichage.

Il faut donc éliminer des programmes pour ZX 81 toutes les instructions FAST et SLOW.

• *Scroll* :

L'instruction SCROLL du ZX 81 permet de faire remonter d'une ligne tout l'écran. Le ZX Spectrum ne possède pas cette instruction. En effet, le message SCROLL ? apparaît dès que l'écran est plein et il faut presser une touche (autre que N ou Break) pour obtenir le défilement de l'écran.

Il faut donc enlever toutes les instructions SCROLL et les remplacer par l'astuce décrite dans le premier chapitre de ce livre (utilisation de la variable système d'adresse 23692). Vous trouverez un exemple d'application de cette astuce dans le jeu KAYAK.

• *Codage des caractères* :

Le ZX 81 utilise un code qui lui est propre alors que le ZX Spectrum a fort heureusement adopté le code ASCII (American Standard Code for Information Interchange).

Il faut donc convertir, chaque fois que c'est nécessaire, les codes du ZX 81 en leurs équivalents ASCII.

• *Screen$, Point, Attr* :

Beaucoup de jeux nécessitent le test du contenu d'une position de l'écran avant l'affichage d'un caractère (test de collision lors du déplacement d'un mobile,...). Le ZX 81 ne possède pas d'instruction particulière pour cela, ce qui oblige le programmeur à utiliser la formule PEEK (PEEK 16398 + 256 * PEEK 16399) pour obtenir le code du caractère affiché à la position courante du curseur.

Le ZX Spectrum est beaucoup plus complet grâce aux instructions SCREEN$, POINT et ATTR.

— SCREEN$(Y,X): renvoie une chaîne contenant le caractère affiché à la position de coordonnées (Y,X) avec $0 \leqslant Y \leqslant 23$ et $0 \leqslant X \leqslant 31$. Cette fonction reconnaît les caractères de codes compris entre 32 et 127, qu'ils soient en mode normal ou inversé. Malheureusement, SCREEN$ ne peut pas être employée pour tester des positions occupées par des caractères redéfinis.

— POINT(X,Y) donne 1 si le pixel de coordonnées (X,Y) avec $0 \leqslant X \leqslant 255$ et $0 \leqslant Y \leqslant 175$, est allumé et donne Ø si ce pixel est éteint.

— ATTR(X,Y) renvoie la valeur contenue dans l'octet attribut pour la position de coordonnées (Y,X) avec $0 \leqslant Y \leqslant 23$ et $0 \leqslant X \leqslant 31$. Cette valeur est calculée par la formule: INK + PAPER * 8 + BRIGHT * 64 + FLASH * 128.

POINT et ATTR seront particulièrement utiles pour les jeux utilisant des caractères redéfinis ou pour tester si une position a des caractéristiques précises.

• *Capacité mémoire*:

Le ZX 81, en version de base, n'a que 1 K de RAM, ce qui contraint les programmeurs à utiliser de nombreuses astuces pour réduire la taille de leurs programmes.

Le ZX Spectrum dispose, en version de base, de 16 K de RAM, donnant ainsi plus de place qu'il n'en faut pour la plupart des jeux.

Vous pourrez donc adapter sans problème les trente cinq jeux du livre "ZX Spectrum à la conquête des jeux". La taille mémoire du ZX Spectrum vous permettra de donner libre cours à votre imagination. Vous aurez certainement des idées pour améliorer des jeux qui, faute de place disponible, ne pouvaient pas être très évolués.

4.2. EXEMPLE D'ADAPTATION D'UN JEU: FIL D'ARIANE

FIL D'ARIANE est un jeu extrait du chapitre Jeux d'Adresse du livre "ZX 81 à la conquête des jeux".

Le principe de ce jeu est très simple : perdu dans un ténébreux labyrinthe, vous êtes amené à suivre un fil salvateur qui conduit à la sortie.

A l'aide des touches 6 et 7, vous devez suivre fidèlement le chemin tracé jusqu'à ce que vous soyez parvenu sur la lettre F du mot FIN.

Les listings du jeu FIL D'ARIANE pour ZX 81 et ZX Spectrum sont données ci-dessous :

Version ZX 81

```
  5 LET B=5
  6 LET C=B+B
  7 LET A=C
 10 PRINT AT A,A-A;"-----"
 20 FOR I=B TO B*B
 30 PRINT AT A,I;"-"
 40 LET A=A+INT (RND*3)-B/B
 50 IF A<B THEN LET A=B
 60 IF A>B+C THEN LET A=B+C
 70 NEXT I
 80 PRINT AT A,I;"FIN"
 90 LET A=C
100 LET I=C-C
110 PRINT AT A,I;"-"
120 LET I=I+B/B
130 LET A=A+(INKEY$="6")-(INKEY
$="7")
140 PRINT AT A,I;
150 LET K=PEEK (PEEK 16396+256*
PEEK 16399)
160 PRINT ">"
163 FOR H=B TO C+B
167 NEXT H
170 IF K=CODE "F" THEN GO TO C*
C+C
180 IF K=CODE "-" THEN GO TO C*
C+C
190 PRINT "VOUS ETES PERDU"
500 PAUSE 4E4
510 CLS
520 RUN
1000 PRINT "BRAVO, VOUS VOILA SA
UVE"
1010 GO TO B*C*C
```

Version ZX Spectrum

```
  5 LET B=5
  6 LET C=B+B
  7 LET A=C
 10 PRINT AT A,A-A;"------"
 20 FOR I=B TO B*B
 30 PRINT AT A,I;"-"
 40 LET A=A+INT (RND*3)-B/B
 50 IF A<B THEN LET A=B
 60 IF A>B+C THEN LET A=B+C
 70 NEXT I
 80 PRINT AT A,I;"FIN"
 90 LET A=C
100 LET I=C-C
110 PRINT AT A,I;"-
120 LET I=I+B/B
130 LET A$=INKEY$
140 LET A=A+(A$="6")-(A$="7")
150 LET K=CODE (SCREEN$ (A,I))
160 PRINT AT A,I; INVERSE 1;">"
163 FOR H=B TO C+B
167 NEXT H
170 IF K=CODE "F" THEN GO TO C*
C*C
180 IF K=CODE "-" THEN GO TO C*
C+C
190 PRINT "VOUS ETES PERDU"
500 PAUSE 4E4
510 CLS
520 RUN
1000 PRINT "BRAVO, VOUS VOILA SA
UVE"
1010 GO TO B*C*C
```

Vous remarquerez que ces deux programmes sont, à peu de choses près, identiques. Seules les lignes 13Ø, 14Ø, 15Ø et 16Ø sont différentes.

Structure du programme

- *Initialisations* : lignes 5 à 1ØØ

Lignes 5 à 7 : initialisations diverses.

Lignes 1Ø à 8Ø : dessin du chemin à suivre.

Lignes 9Ø à 1ØØ : initialisation de la position de départ.

- *Boucle de jeu* : lignes 11Ø à 18Ø

Ligne 11Ø : redessine le chemin après le passage du mobile.

Ligne 12Ø : avance horizontale.

Ligne 13Ø (et ligne 14Ø pour ZX Spectrum) : calcul déplacement vertical.

Ligne 15Ø (et ligne 14Ø pour ZX 81) : sauvegarde le code du caractère placé sur la prochaîne position d'affichage.

Ligne 16Ø : affichage du mobile.

Lignes 163 et 167 : boucle de temporisation destinée à ralentir le déplacement du mobile.

Ligne 17Ø : sortie du labyrinthe.

Ligne 18Ø : retour au début de la boucle de jeu tant que le mobile reste sur le chemin.

- *Fin de la partie* : lignes 19Ø à 1Ø1Ø

Ligne 19Ø : le mobile est sorti du chemin ; vous êtes perdu.

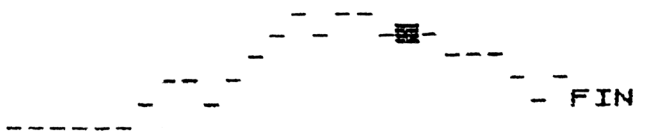
Ligne 5ØØ : attend qu'une touche soit pressée pour continuer.

Ligne 51Ø : effacement de l'écran.

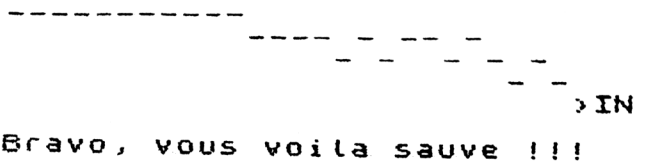
Ligne 52Ø : début de nouvelle partie.

Ligne 1ØØØ : le mobile est sorti du "labyrinthe".

Ligne 1Ø1Ø : retour en ligne 5ØØ.



Adapter un jeu, c'est bien ; l'améliorer, c'est mieux. Vous trouverez ci-dessous une version pour ZX Spectrum du jeu FIL D'ARIANE. Le programme utilise les possibilités du micro-ordinateur en matière de sons et de couleurs. La notion de niveau de jeu a été introduite afin de pouvoir régler la difficulté du jeu (niveau 1 : déplacement lent, niveau 1Ø : déplacement très rapide).



Structure du programme

- *Programme principal* : lignes 1 à 9Ø

Ligne 1Ø : appel du sous-programme de présentation.

Ligne 2Ø : appel du sous-programme d'initialisation.

Ligne 3Ø : appel du sous-programme de jeu.

Ligne 4Ø : affichage clignotant du mobile en fin de partie. Appel d'une routine de sonorisation.

Ligne 5Ø : cas où vous êtes perdu...

Ligne 6Ø : cas où vous êtes sauvé !!

Ligne 7Ø : attente de la pression d'une touche pour continuer.

Ligne 8Ø : effacement de l'écran.

Ligne 9Ø : retour au début du jeu.

- *Sous-programme de jeu* : lignes 5ØØØ à 51ØØ

Ligne 5ØØØ : redessine le chemin après le passage du mobile.

Ligne 5Ø1Ø : avance horizontale.

Ligne 5Ø2Ø : lecture du clavier.

Ligne 5Ø3Ø : calcul déplacement vertical.

Ligne 5Ø4Ø : sauvegarde du caractère situé sur la prochaine position d'affichage.

Ligne 5Ø5Ø : affichage du mobile.

Ligne 5Ø6Ø : boucle de temporisation dont la durée dépend du jeu choisi.

Ligne 5Ø7Ø : sortie victorieuse du labyrinthe signalée par le drapeau FIN à zéro.

Ligne 5Ø8Ø : vérifie si le mobile reste bien sur le chemin.

Ligne 5Ø9Ø : mise à 1 du drapeau FIN si le mobile a perdu le fil...

Ligne 51ØØ : retour au programme principal.

- *Sous-programme d'initialisation* : lignes 8ØØØ à 814Ø

Lignes 8ØØØ à 8Ø2Ø : choix du niveau de jeu.

Ligne 8Ø3Ø : effacement de l'écran.

Ligne 8Ø4Ø à 812Ø : affichage du chemin à suivre en prenant soin à ce que le début soit en ligne droite (ligne 8Ø4Ø).

Ligne 813Ø : coordonnées de la position de départ.

Ligne 814Ø : retour au programme principal.

- *Sous-programme de présentation* : lignes 9ØØØ à 912Ø

Ligne 9ØØØ : l'écran et son bord sont de couleur noire ; les caractères seront verts.

Ligne 9Ø1Ø : affichage du nom du jeu.

Ligne 9Ø2Ø : affichage d'un chemin.

Lignes 9Ø3Ø à 9Ø7Ø : déplacement d'un mobile le long du chemin dessiné par la ligne 9Ø2Ø.

Ligne 9Ø75 : affichage clignotant du mobile.

Lignes 9Ø8Ø à 911Ø : routine de sonorisation.

Ligne 912Ø : retour au programme principal.

```
  1 REM ****************
  2 REM * FIL D'ARIANE *
  3 REM ****************
 10 GO SUB 9000
 20 GO SUB 8000
 30 GO SUB 5000
 40 PRINT AT y,x; BRIGHT 1; FLA
SH 1; INK 2;">": GO SUB 9080
 50 IF fin=1 THEN PRINT AT 16,5
; FLASH 1;" Vous etes perdu !!!
": GO TO 70
 60 PRINT AT 16,3; FLASH 1;" Br
avo, vous voila sauve !!! "
 70 PAUSE 4E4: PAUSE 4E4
 80 CLS
 90 GO TO 20
5000 PRINT AT y,x;"-"
5010 LET x=x+1
5020 LET a$=INKEY$
5030 LET y=y+(a$="6")-(a$="7")
5040 LET k$=SCREEN$ (y,x)
5050 PRINT AT y,x; INVERSE 1; IN
K 2;">"
5060 FOR j=1 TO 10*(10-n): NEXT
j: BEEP 0.01,10
5070 IF k$="F" THEN LET fin=0: G
O TO 5100
5080 IF k$="-" THEN GO TO 5000
5090 LET fin=1
5100 RETURN
```

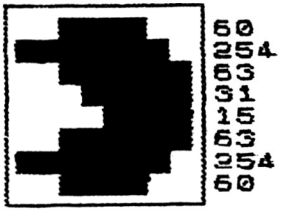
```
8000 INPUT "Niveau de jeu (1-10)
 ?";n
8010 LET n=INT (ABS n)
8020 IF (n-1)*(n-10)>0 THEN GO T
O 8000
8030 CLS
8040 PRINT AT 10,0;"-----"
8050 LET y=10
8060 FOR x=5 TO 25
8070 PRINT AT y,x;"-"
8080 LET y=y+INT (RND*3)-1
8090 IF y<5 THEN LET y=5
8100 IF y>15 THEN LET y=15
8110 NEXT x
8120 PRINT AT y,x;"FIN"
8130 LET x=0: LET y=10
8140 RETURN
9000 BORDER 0: PAPER 0: INK 4: C
LS
9010 PRINT AT 10,10; FLASH 1; IN
K 5;"FIL D'ARIANE"
9020 PRINT AT 20,0;"------------
--------------FIN"
9030 FOR i=0 TO 25
9040 PRINT AT 20,i; BRIGHT 1; FL
ASH 1; INK 2;">"
9050 BEEP 0.01,20
9060 PRINT AT 20,i;"-"
9070 NEXT i
9075 PRINT AT 20,i; BRIGHT 1; FL
ASH 1; INK 2;">"
9080 FOR i=10 TO 30
9090 BEEP 0.01,i
9100 BEEP 0.01,2*i-20
9110 NEXT i
9120 RETURN
```

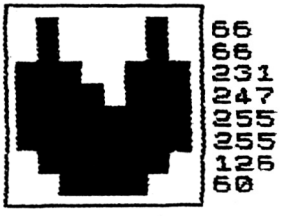
Annexe 1
L'écran du ZX Spectrum

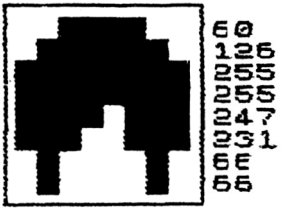
Illustration extraite du manuel Sinclair.

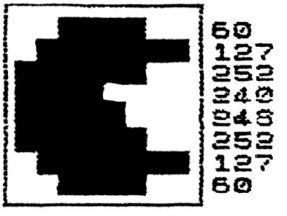
Annexe 2
Bibliothèque de caractères

- *PICMAN*

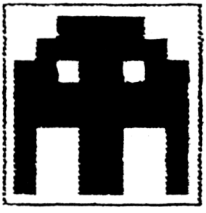


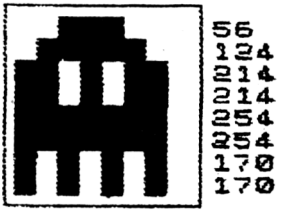




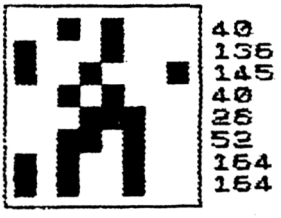


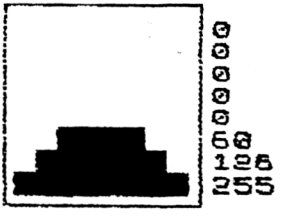
- *Monstres*



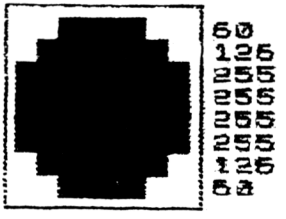


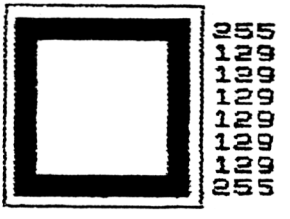
- *Décomposition de PICMAN*



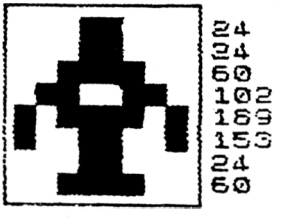


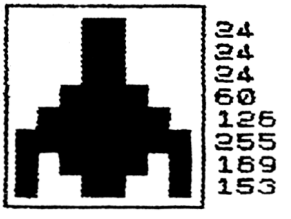
- *Pions*



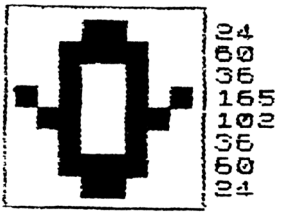


- *Fusées*

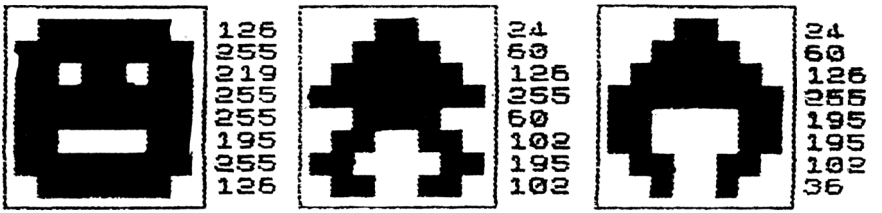




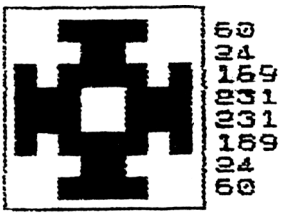
- *Canöe*



- *Envahisseurs*



- *Chenille, chase*



Imprimerie de la Manutention à Mayenne
Dépôt légal : novembre 1984
N° d'Éditeur : 4172

Découvrez avec ZX SPECTRUM A LA CONQUETE DES JEUX l'univers fantastique des jeux sur micro-ordinateur.

Le premier chapitre vous apprendra les principales techniques de programmation des jeux en Basic sur ZX Spectrum ainsi que des astuces très utiles pour créer des «sprites», utiliser les manettes de jeu, redéfinir le générateur de caractères, employer certaines variables «système» afin d'obtenir des effets spéciaux. Vous trouverez en particulier le programme DESSIN qui est un outil très puissant et très simple d'emploi pour définir des caractères.

Le second chapitre vous dévoilera quelques trucs indispensables pour programmer des jeux en langage machine sur ZX Spectrum : affichage de caractères, lecture du clavier, accès au générateur de sons, ...

Dix jeux de réflexion, d'action et d'aventure vous sont ensuite proposés. Chaque programme est accompagné d'une analyse détaillée de sa structure. Vous pourrez ainsi acquérir des connaissances nouvelles en programmation pour écrire ensuite vos propres jeux.

Tous les jeux, sauf EXPLORATION SPATIALE, sont destinés au ZX Spectrum 16 K. Saurez-vous éviter les monstres diaboliques qui poursuivent le pauvre PICMAN, provoquer votre adversaire à la faute dans CHASE ou guider votre KAYAK le plus longtemps possible dans un torrent déchaîné ?

EXPLORATION SPATIALE est un jeu d'aventure très complexe que sa longueur réserve au ZX Spectrum 48 K. Vous devez partir à la recherche d'une atompile, qui vous permettra de faire décoller votre vaisseau spatial de cette maudite planète où la «malchance» vous a conduit. Votre route sera longue et difficile. La faim vous tenaillera très vite et de nombreux dangers vous guettent !

Inutile de dévoiler plus longuement les secrets de ce monde fascinant. Découvrez-les vous-même pour votre plus grand plaisir.

EYROLLES

8647

EYROLLES

ZX SPECTRUM A LA CONQUÊTE DES JEUX

A. PERBOST